Mathematics and Computation
7'th edition
Volume 1

Klaus Grue

June 1, 2001

Mathematics and Computation, 7'th edition, Volume 1 of 3
Klaus Grue, DIKU, Universitetsparken 1, DK-2100 Copenhagen, Denmark
grue@diku.dk, http://www.diku.dk/~grue/
©1994–2001 Klaus Grue

ISBN 87-981270-3-9

Contents

Chapter 1

Syntax

1.1 The Map system

$[\, f_1(\mathsf{x}) \doteq \mathsf{x}^2 - 1 \,]$ is a definition and $[\, f_1(10) = 99 \,]$ is a true statement. You can use the definition to verify the statement. The verification is so simple, that even a computer could do it.

Well, actually a computer *has* done it. Before you got this book in your hand, it was proofread by Map. Map is a computer program that can read books, understand definitions, verify simple mathematical claims, and generate computer code. You can obtain a copy of Map and use it for mathematics and/or programming.

This book contains no formal errors. I can say so, because it was proofread by Map. Of course, the book may contain informal errox like any other book, but $[\, f_1(10) = 99 \,]$ is a formal claim that you'd better believe.

(Note for 2001/02 version: Map is not yet operative, sorry)

1.2 How Map treats formulas

You need to know a little about Map in order not to be puzzled by the various brackets in this book.

Map checks the form and contents of text in brackets but the amount of checking can be controlled by small decorations. As an example, Map accepts $[\, f_1(10) = 98 \,]^{\cdot}$ because it has the form of a statement. Don't remove the little dot superscript of the right bracket; that would make Map complain because $[\, f_1(10) = 98 \,]^{\cdot}$ is false. As another example, Map accepts $[\, = \,]^{\circ}$ because of the circle superscript. Map simply skips anything inside brackets that have a circle superscript.

Map does not care about the order of statements in a text. Later in this book, you will see the definitions of $[\, \doteq \,]^{\circ}$, $[\, = \,]^{\circ}$, $[\, - \,]^{\circ}$, and exponentiation. Further, you will see that $[\, \mathsf{x} \,]^{\cdot}$ is classified as a variable and that $[\, 01 \ldots 9 \,]^{\cdot}$ are classified as digits. Since Map makes no assumptions about the alphabet, you can use it

equally well for e.g. English, Greek and Russian. Even the error messages from Map are declared in the text and can be translated to any language.

1.3 Index

All words in *italics* occur in the *index* at the end of this book, so whenever you see italics, remember that you can find back to this particular place using the index. As an example, if you look up "index" in the index, you will find a reference to the page you read right now.

All words in italics also occur as a footnote. As an example, you may have noted two footnotes that say "italics" and "index". After reading a chapter, you can get a fairly precise summary by scanning through these footnotes.

Occasionally, my text processing system places footnotes one page away from what they refer to. I'm sorry.

1.4 How I read formulas

I prefer to read $\boxed{x+y}$ as "x plus y" and $\boxed{x \cdot y}$ as "x times y". As an example, I prefer to read $2 \cdot x + 3$ thus:

two times x plus three

How you read formulas is up to you, we got freedom of speech. However, you may find it convenient to know how I read formulas if you want to look up a particular construct in the index of this book.

Formulas in boxes occur in the index just like words in italics. $x + y$ is boxed above, so you can find it in the index.

Whenever I introduce mathematical notation that you can look up in the index, I also state in a footnote how I read the construct. As an example, you may have noted a footnote that says "$x+y$ x plus y". That footnote indicates that you can look up $x + y$ under "x plus y" in the index, i.e. under "x" in the index. You can also find $x + y$ in the math section of the index which is the section before "A".

I hope one day to be able to read formulas into a microphone and leave it to a computer to typeset what I read. While I write this book, however, speech recognition technology has not yet had its break through.

I will use the name $\boxed{\text{Pük}}$ for the language I use for reading formulas. I constructed the name Pük by removing "Vola" from "Volapük". Volapük was the first, artificial, spoken language. It was constructed by one man and only spoken by him and his wife. The man refused to speak any other language. The

---

italics
index
$x + y$ x plus y
$x \cdot y$ x times y
Pük pyk

only word from Volapük that has entered other languages is the word "Volapük", which is the name of the language in the language itself. In Volapük, "vola" means "world" and "pük" means "speak", so Volapük means "world speak".

Pük is intended to be spoken, but can also be written as done above. Written Pük only contains small letters and does not contain any punctuation marks as capital letters and punctuation marks are difficult to enter through a microphone.

I prefer to read $\boxed{\boxed{0}}^{\cdot}$, $\boxed{\boxed{1}}^{\cdot}$, $\boxed{\boxed{2}}^{\cdot}$, $\boxed{\boxed{3}}^{\cdot}$, $\boxed{\boxed{4}}^{\cdot}$, $\boxed{\boxed{5}}^{\cdot}$, $\boxed{\boxed{6}}^{\cdot}$, $\boxed{\boxed{7}}^{\cdot}$, $\boxed{\boxed{8}}^{\cdot}$, and $\boxed{\boxed{9}}^{\cdot}$ as "null", "one", "two", "three", "four", "five", "six", "seven", "eight", and "nine", respectively.

I prefer to read variable names like $\boxed{\boxed{\mathsf{a}}}^{\cdot}$, $\boxed{\boxed{\mathsf{A}}}^{\cdot}$, and $\boxed{\boxed{\mathcal{A}}}^{\cdot}$ as "a", "capital a", and "script a", respectively, and I do likewise for the remaining 25 letters of the English alphabet.

I prefer to read $\boxed{\boxed{\mathsf{x}-\mathsf{y}}}^{\cdot}$ as "x minus y" and $\boxed{\boxed{\mathsf{x}^{\mathsf{y}}}}^{\cdot}$ as "x power y end".

Exercise 1.4.1 Translate this to Pük:

(a) $\boxed{\mathsf{x}^{\mathsf{y}+2}}^{\cdot}$

(b) $\boxed{\mathsf{x}^{\mathsf{y}}+2}^{\cdot}$

(c) $\boxed{\mathsf{x}^2+3}^{\cdot}$.

(d) $\boxed{\mathsf{a}^{\mathsf{A}}+\mathcal{A}}^{\cdot}$.

You can find answers to exercises at the end of each chapter.

1.5 Computation

I assume you see no problem in computing $\boxed{2\cdot 3+4}^{\cdot}$. Computation of $\boxed{2\cdot 3+4}^{\cdot}$ is not at all simple, however. It is just that you and I have been trained so much in such computations that we don't see the problems anymore. I will go through

$\boxed{0}^{\cdot}$	null
$\boxed{1}^{\cdot}$	one
$\boxed{2}^{\cdot}$	two
$\boxed{3}^{\cdot}$	three
$\boxed{4}^{\cdot}$	four
$\boxed{5}^{\cdot}$	five
$\boxed{6}^{\cdot}$	six
$\boxed{7}^{\cdot}$	seven
$\boxed{8}^{\cdot}$	eight
$\boxed{9}^{\cdot}$	nine
$\boxed{\mathsf{a}}^{\cdot}$	a
$\boxed{\mathsf{A}}^{\cdot}$	capital a
$\boxed{\mathcal{A}}^{\cdot}$	script a
$\boxed{\mathsf{x}-\mathsf{y}}^{\cdot}$	x minus y
$\boxed{\mathsf{x}^{\mathsf{y}}}^{\cdot}$	x power y end

the computation in quite a lot of detail because this example reveals some of the general principles of computation. The computation goes like this:

$$[\![\, 2 \cdot 3 + 4 \equiv 6 + 4 \equiv 10 \,]\!].$$

As you can see, the computation falls in two steps. In the first step I replace $[\![\, 2 \cdot 3 \,]\!]$ by $[\![\, 6 \,]\!]$ and in the second step I add $[\![\, 6 \,]\!]$ and $[\![\, 4 \,]\!]$.

The fact that $[\![\, 2 \cdot 3 \equiv 6 \,]\!]$ is why I can replace $[\![\, 2 \cdot 3 \,]\!]$ by $[\![\, 6 \,]\!]$ in the first step. The literature calls this *substitution of equals*. Substitution of equals says that one can replace $[\![\, 2 \cdot 3 \,]\!]$ by $[\![\, 6 \,]\!]$ in any expression without affecting the value of that expression because $[\![\, 2 \cdot 3 \equiv 6 \,]\!]$. More generally, substitution of equals says:

> If $[\![\, \mathcal{A} \equiv \mathcal{B} \,]\!]$, then one can replace $[\![\, \mathcal{A} \,]\!]$ by $[\![\, \mathcal{B} \,]\!]$ in any expression without affecting the value of that expression.

In the first step of the computation above, $[\![\, \mathcal{A} \,]\!]$ is $[\![\, 2 \cdot 3 \,]\!]$, $[\![\, \mathcal{B} \,]\!]$ is $[\![\, 6 \,]\!]$, and the expression is $[\![\, 2 \cdot 3 + 4 \,]\!]$ before and $[\![\, 6 + 4 \,]\!]$ after the substitution.

Exercise 1.5.1 Compute $[\![\, 1 + 2 + 3 + 4 \,]\!]$. How many steps do you have to make?

1.6 Equal signs

Until now I have used three different equal signs: $[\![\, \equiv \,]\!]^\circ$, $[\![\, \doteq \,]\!]^\circ$ and $[\![\, = \,]\!]^\circ$. $\Big[\!\Big[\, \boxed{\mathcal{A} \equiv \mathcal{B}} \,\Big]\!\Big]$ means that $[\![\, \mathcal{A} \,]\!]$ and $[\![\, \mathcal{B} \,]\!]$ are equal, $\Big[\!\Big[\, \boxed{\mathcal{A} \doteq \mathcal{B}} \,\Big]\!\Big]$ means that I define $[\![\, \mathcal{A} \,]\!]$ to be equal to $[\![\, \mathcal{B} \,]\!]$, and I use $\Big[\!\Big[\, \boxed{\mathcal{A} = \mathcal{B}} \,\Big]\!\Big]$ when I ask Map to compute $[\![\, \mathcal{A} \,]\!]$ and $[\![\, \mathcal{B} \,]\!]$ to compare them for equality. Hence, $\Big[\!\Big[\, \boxed{f_1(\mathsf{x})} \doteq \mathsf{x}^2 - 1 \,\Big]\!\Big]$ defines $[\![\, f_1(x) \,]\!]$ to be equal to $[\![\, x^2 - 1 \,]\!]$, $[\![\, f_1(2) \equiv 3 \,]\!]$ states that $[\![\, f_1(2) \,]\!]$ equals $[\![\, 3 \,]\!]$, and $[\![\, f_1(2) = 3 \,]\!]$ asks Map to verify that $[\![\, f_1(2) \,]\!]$ equals $[\![\, 3 \,]\!]$. As you will see, I will use many other equal signs as well, and they all differ from each other.

I give an overview of equal signs in Section 1.18. See Section 2.11, Section 2.12, Section 2.14, and Section 3.4 for examples of differences between $[\![\, \mathcal{A} \equiv \mathcal{B} \,]\!]$ and $[\![\, \mathcal{A} = \mathcal{B} \,]\!]$.

1.7 Priority

There is more to computation of $[\![\, 2 \cdot 3 + 4 \,]\!]$ than meets the eye. Consider the following argument: $[\![\, 3 + 4 \,]\!]$ is $[\![\, 7 \,]\!]$, hence $[\![\, 2 \cdot 3 + 4 \equiv 2 \cdot 7 \,]\!]$ since $[\![\, 2 \cdot 7 \,]\!]$ arises from $[\![\, 2 \cdot 3 + 4 \,]\!]$ by replacing $[\![\, 3 + 4 \,]\!]$ by $[\![\, 7 \,]\!]$. This gives rise to the faulty computation $[\![\, 2 \cdot 3 + 4 \equiv 2 \cdot 7 \equiv 14 \,]\!]$.

---

	substitution of equals
$[\![\, \mathsf{x} \equiv \mathsf{y} \,]\!]$	x equal y
$[\![\, \mathsf{x} \doteq \mathsf{y} \,]\!]$	x defined equal y
$[\![\, \mathsf{x} = \mathsf{y} \,]\!]$	x computational equal y
$[\![\, f_1(\mathsf{x}) \,]\!]$	f one of x end

$[\,2\cdot 3+4\equiv 6+4\equiv 10\,]^{\cdot}$ is right and $[\,2\cdot 3+4\equiv 2\cdot 7\equiv 14\,]^{\cdot}$ is wrong due to the convention that $[\,\cdot\,]^{\circ}$ has higher *priority* than $[\,+\,]^{\circ}$. This convention means that $[\,\cdot\,]^{\circ}$ has to be computed before $[\,+\,]^{\circ}$ so that $[\,2\cdot 3+4\equiv 6+4\equiv 10\,]^{\cdot}$ is right. Another way to formulate this is that $[\,2\cdot 3+4\,]^{\cdot}$ tacitly means $[\,(2\cdot 3)+4\,]^{\cdot}$ where the parenthesis indicates the order of computation. I prefer the second formulation because it leaves valid the rule of substitution of equals: One cannot replace $[\,3+4\,]^{\cdot}$ by $[\,7\,]^{\cdot}$ in $[\,(2\cdot 3)+4\,]^{\cdot}$ simply because $[\,3+4\,]^{\cdot}$ does not occur anywhere in $[\,(2\cdot 3)+4\,]^{\cdot}$.

$[\,x\cdot y\,\boxed{\breve{>}}\,x+y\,]$ tells Map that $[\,\cdot\,]^{\circ}$ has higher priority than $[\,+\,]^{\circ}$. This allows Map to read $[\,2\cdot 3+4\,]^{\cdot}$ as $[\,(2\cdot 3)+4\,]^{\cdot}$. To verify $[\,2\cdot 3+4=10\,]$, Map needs to know the definitions of $[\,+\,]^{\circ}$ and $[\,\cdot\,]^{\circ}$ as well as the priority rule $[\,x\cdot y\,\breve{>}\,x+y\,]$. I state the definitions of $[\,+\,]^{\circ}$ and $[\,\cdot\,]^{\circ}$ later.

See Appendix A.2 for a complete collection of priority rules.

Exercise 1.7.1 Insert the tacit parentheses in the following expressions and compute their values:

(a) $[\,4+5\cdot 6\,]^{\cdot}$

(b) $[\,2\cdot 3+4\cdot 5\,]^{\cdot}$

(c) $[\,(2+3)\cdot 4\,]^{\cdot}$

1.8 Arity and fixity

There is more to the computation of $[\,2\cdot 3+4\,]^{\cdot}$ than the priority of $[\,+\,]^{\circ}$ and $[\,\cdot\,]^{\circ}$. You need to know e.g. that $[\,\cdot\,]^{\circ}$ is a "binary infix operator". I explain "binary", "infix", and "operator" in the following.

A mathematical *operator* is something that takes some input and produces a value. In $[\,2\cdot 3\,]^{\cdot}$, $[\,\cdot\,]^{\circ}$ takes two pieces of input, namely $[\,2\,]^{\cdot}$ and $[\,3\,]^{\cdot}$, and then computes the product of the two pieces. A piece of input is called an *argument*. In $[\,2\cdot 3\,]^{\cdot}$, $[\,\cdot\,]^{\circ}$ takes two arguments, namely $[\,2\,]^{\cdot}$ and $[\,3\,]^{\cdot}$.

An operator is *unary* if it takes one argument, *binary* if it takes two, and *ternary* if it takes three. An operator that takes $[\,117\,]^{\cdot}$ arguments is $[\,117\,]^{\cdot}$-*ary*. Self-sufficient operators that take no arguments are *null-ary*. A number such as $[\,2\,]^{\cdot}$ or $[\,3\,]^{\cdot}$ takes no arguments, so numbers are examples of null-ary operators. Variables like $[\,x\,]^{\cdot}$ and $[\,y\,]^{\cdot}$ are null-ary as well.

---

	priority
$[\,x\,\breve{>}\,y\,]^{\cdot}$	x greater priority y
	operator
	argument
	unary
	binary
	ternary
	ary, $[\,117\,]^{\cdot}$-
	null-ary

The number of arguments of an operator is known as the *arity* of the operator. As examples, the arity of $[\,\cdot\,]^\circ$ is $[\,2\,]^\cdot$ and the arity of $[\,x\,]^\cdot$ is $[\,0\,]^\cdot$.

An operator is *prefix* if it occurs before its arguments, *suffix* if it occurs after its arguments, *infix* if it occurs between the arguments, and *outfix* if it engulfs its arguments. I refer to this aspect of an operator as its *fixity*. Constructs like $\left[\,\frac{x}{y}\,\right]^\cdot$ and $[\,x^y\,]^\cdot$ are too complicated to have a fixity. The terms "prefix", "suffix" and "infix" are commonly known terms. I'm afraid "outfix" and "fixity" are my responsibility. Here you have some examples:

Operator	Arity	Fixity
$[\,2\,]^\cdot$	null-ary	outfix
$[\,x\cdot y\,]^\cdot$	binary	infix
$[\,x+y\,]^\cdot$	binary	infix
$[\,-x\,]^\cdot$	unary	prefix
$[\,x!\,]^\cdot$	unary	suffix
$[\,\text{if } x \text{ then } y \text{ else } z\,]^\cdot$	ternary	prefix
$[\,f_1(x)\,]^\cdot$	unary	outfix
$\left[\,\frac{x}{y}\,\right]^\cdot$	binary	none
$[\,x^y\,]^\cdot$	binary	none

If you want to find the fixity of an operator, do as follows: If the operator is a variable, then the fixity is outfix. Otherwise, write a term that consists of the operator applied to variables. Then look at the first and last character of the term to see whether or not that character is a variable. Finally, look up the fixity in the table below.

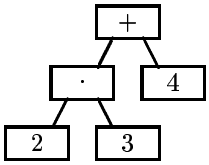
First character	Last character	Fixity
variable	variable	infix
variable	not a variable	suffix
not a variable	variable	prefix
not a variable	not a variable	outfix

As you can see, you need to know a lot of conventions just to compute $[\,2\cdot 3+4\,]^\cdot$. You need to know that $[\,2\,]^\cdot$, $[\,3\,]^\cdot$, and $[\,4\,]^\cdot$ are null-ary outfix operators whereas $[\,\cdot\,]^\circ$ and $[\,+\,]^\circ$ are binary infix operators, and you need to know that $[\,\cdot\,]^\circ$ has greater priority than $[\,+\,]^\circ$. And then, of course, you must know how to add and multiply.

1.9 Syntax trees

Here you have another way to write $[\,2\cdot 3+4\,]^\cdot$:

arity
prefix
suffix
infix
outfix
fixity

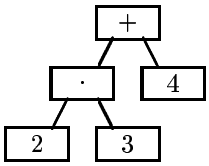


I will call this a *syntax tree*.

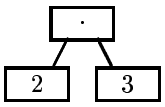
The syntax tree above consists of five *nodes* and four *edges*. The nodes are the boxes and the edges are the lines between the boxes. Each node has a *label* which is what is written inside the box.

The node at the top is called the *root node* of the syntax tree. (Natural trees grow upwards from the root but syntax trees seem to grow downwards from the root node throughout the literature, c.f. [2], p.305). The label of the root node is simply called the *root* of the tree, so $[\,+\,]^\circ$ is the root of the above syntax tree.

Two edges extend downwards from the root node, and there is a *subtree* attached to each of these edges. I number these subtrees from left to right so that the first subtree of



is



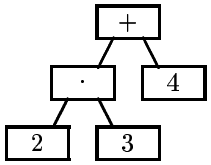
and the second is

The root of the first subtree is $[\,\cdot\,]^\circ$ and the first subtree in turn has two subtrees. The root of the second subtree is $[\,4\,]^\cdot$ and the second subtree has no subtrees. The root of the first subtree of the first subtree is $[\,2\,]^\cdot$ and the root of the second subtree of the first subtree is $[\,3\,]^\cdot$.

The syntax tree

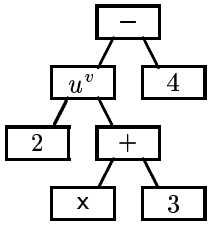
---

syntax tree
node
edge
label
root node
root
subtree



brings out the structure of $\lceil\, 2 \cdot 3 + 4 \,\rceil$ very clearly. I prefer to think of syntax trees as the true mathematical notation, and I prefer to think about $\lceil\, 2 \cdot 3 + 4 \,\rceil$ as a highly compact way of representing a syntax tree.

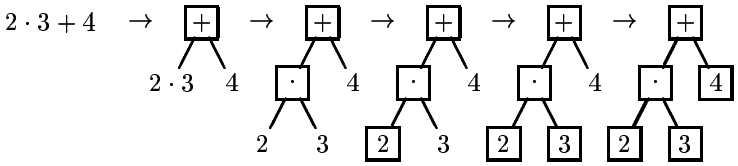
Here you have the syntax tree of $\lceil\, 2^{x+3} - 4 \,\rceil$:



1.10 Stepwise formation of term trees

The *principal operator* of an expression is the operator that you compute last. If you compute $\lceil\, 2 \cdot 3 + 4 \,\rceil$, then you compute $\lceil\, + \,\rceil^\circ$ last, so $\lceil\, + \,\rceil^\circ$ is the principal operator of $\lceil\, 2 \cdot 3 + 4 \,\rceil$.

The principal operator of an expression is the same as the root of the corresponding syntax tree. You can use this to convert an expression into a syntax tree. Here you have a stepwise conversion of $\lceil\, 2 \cdot 3 + 4 \,\rceil$ into a syntax three:



In the first step of the conversion, I spot $\lceil\, + \,\rceil^\circ$ as the principal operator of $\lceil\, 2 \cdot 3 + 4 \,\rceil$, so I put $\lceil\, + \,\rceil^\circ$ in the root of the syntax tree. $\lceil\, + \,\rceil^\circ$ is binary, so I draw two downward edges from the root. The arguments of $\lceil\, + \,\rceil^\circ$ are $\lceil\, 2 \cdot 3 \,\rceil$ and $\lceil\, 4 \,\rceil$, so I write these expressions at the ends of the corresponding edges. I take care to keep the left to right ordering of the arguments: I put $\lceil\, 2 \cdot 3 \,\rceil$ to the left and $\lceil\, 4 \,\rceil$ to the right.

The second step is exactly like the first step except that I work on $\lceil\, 2 \cdot 3 \,\rceil$.

In the third step I realise that $\lceil\, 2 \,\rceil$ is a null-ary operator, so I just put a box around it.

The fourth and fifth steps are similar to the third one.

principal operator

Exercise 1.10.1 Draw the syntax trees of the following expressions and compute their values.

(a) $[\, 5 \cdot 1 + 8 \,]$

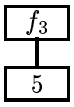
(b) $[\, 2 + 3 \cdot 5 \,]$

(c) $[\, 1 \cdot 2 + 3 \cdot 4 \,]$

(d) $[\, 8 \,]$

1.11 Definitions

$\left[\, \boxed{f_3(\mathsf{x})} \doteq 2 \cdot \mathsf{x} + 4 \,\right]$ defines a new operator $[\, f_3(*) \,]$. $[\, f_3(\mathsf{x}) \,]$ is unary in the sense that it operates on a single subexpression $[\, \mathsf{x} \,]$. As an example, the syntax tree of $[\, f_3(5) \,]$ is:



You have $[\, f_3(5) \equiv 14 \,]$. The syntax tree of $[\, 14 \,]$ reads:

As you can see, $[\, f_3(5) \,]$ and $[\, 14 \,]$ are equal (meaning that they have equal value) even though they are not *identical* (i.e. they are different expressions). Expressions that are equal without being identical is what mathematics is all about.

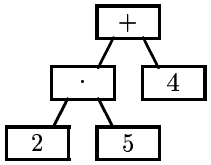
Usually, a mathematician thinks of $[\, f_3(5) \,]$ as a value rather than as an expression. Hence, $[\, f_3(5) \,]$ and $[\, 14 \,]$ are equal to a mathematician because they denote the same value. The phenomenon that a mathematician writes an expression but thinks of a value is called *referential transparency* in the literature.

You also have referential transparency outside mathematics. If I say 'Ulysses blinded Polyphemos' then you probably think of 'Ulysses' as a (possibly mythical) person and 'Polyphemos' as a (likely mythical) cyclops, even though 'Ulysses' and 'Polyphemos' are really just names.

1.12 Macro definitions

$\left[\, f_4(\mathsf{x}) \,\boxed{\doteq}\, 2 \cdot \mathsf{x} + 4 \,\right]$ does not define $\left[\, \boxed{f_4(\mathsf{x})} \,\right]$ as a new operator but merely introduces $[\, f_4(\mathsf{x}) \,]$ as shorthand for $[\, 2 \cdot \mathsf{x} + 4 \,]$. Hence, the syntax tree of $[\, f_4(5) \,]$ reads

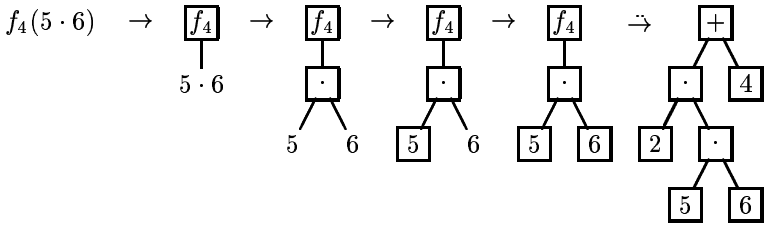
$[\, f_3(\mathsf{x}) \,]$	f three of x end
	identical
	referential transparency
$[\, \mathsf{x} \doteq \mathsf{y} \,]$	x macro equal y
$[\, f_4(\mathsf{x}) \,]$	f four of x end



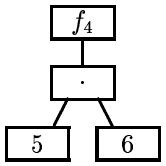
because $[\![f_4(5)]\!]$ is just shorthand for $[\![2 \cdot 5 + 4]\!]$. In general, $[\![\doteq]\!]^\circ$ introduces new operators and $[\![\ddot{=}]\!]^\circ$ introduces shorthand notation.

I will refer to constructs like $[\![f_4(\mathsf{x})]\!]$ that are defined using $[\![\ddot{=}]\!]^\circ$ as *macros* (as opposed to *operators*). I will refer to definitions like $[\![f_4(\mathsf{x}) \ddot{=} 2 \cdot \mathsf{x} + 4]\!]$ that use $[\![\ddot{=}]\!]^\circ$ as *macro definitions*.

Here you have the stepwise conversion of $[\![f_4(5 \cdot 6)]\!]$ into a syntax tree:



The first four steps convert the term into a tree like you saw it in Section 1.10. I will refer to such steps as *parse reductions*. The result of performing parse reductions is the *parse tree*. The parse tree of $[\![f_4(5 \cdot 6)]\!]$ is:

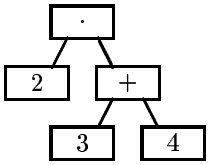


The fifth step in finding the syntax tree of $[\![f_4(5 \cdot 6)]\!]$ is a *macro reduction*. In the macro reduction, $[\![f_4]\!]^\circ$ is replaced by the tree it is shorthand for. In general, to find the syntax tree of an expression you first perform parse reductions until no further parse reductions are possible. After that, you perform macro reductions until no further macro reductions are possible.

1.13 Parentheses

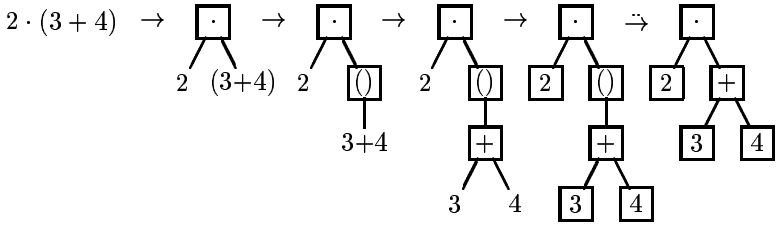
$[\![\boxed{(\mathsf{x})} \ddot{=} \mathsf{x}]\!]$ is a very important use of $[\![\doteq]\!]^\circ$. It tells Map that $[\![(2)]\!]$ denotes the same syntax tree as $[\![2]\!]$ and $[\![((3) + (4))]\!]$ denotes the same syntax tree as $[\![3 + 4]\!]$. As another example you have that the syntax tree of $[\![2 \cdot (3 + 4)]\!]$ is

	macro
	operator
	macro definition
	parse reduction
	parse tree
	macro reduction
$[\![(\mathsf{x})]\!]$	parenthesis x end

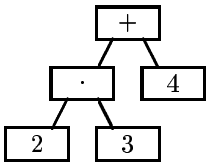


As you can see, the parentheses do not show up in the syntax tree.

Here you have the stepwise conversion of $\lceil\, 2 \cdot (3+4) \,\rceil^{\cdot}$ into a syntax tree:



The syntax trees of $\lceil\, 2 \cdot 3 + 4 \,\rceil^{\cdot}$ and $\lceil\, (2 \cdot 3) + 4 \,\rceil^{\cdot}$ are both:



Exercise 1.13.1 Draw the syntax trees of the following expressions and compute their values.

a $\lceil\, 2 \cdot 3 + (4 \cdot 5) \,\rceil^{\cdot}$

b $\lceil\, f_3(2+3) \,\rceil^{\cdot}$

c $\lceil\, f_4(2+3) \,\rceil^{\cdot}$

1.14 Associativity

$\lceil\, 5 - 2 - 1 \,\rceil^{\cdot}$ could mean either $\lceil\, (5-2) - 1 \,\rceil^{\cdot}$ or $\lceil\, 5 - (2-1) \,\rceil^{\cdot}$. However, $\lceil\, \mathsf{x-y-z} \overset{\smile}{\rightarrow} \mathsf{(x-y)-z} \,\rceil$ tells Map that $\lceil\, (5-2)-1 \,\rceil^{\cdot}$ is the right interpretation. $\lceil\, \mathsf{x-y-z} \overset{\smile}{\rightarrow} \mathsf{(x-y)-z} \,\rceil$ is an example of an *associativity rule*. The literature calls $\lceil\, \mathsf{x-y} \,\rceil^{\cdot}$ *left associative* because $\lceil\, 5-2-1 \,\rceil^{\cdot}$ has the tacit parenthesis around the leftmost $\lceil\, - \,\rceil^{\circ}$. Operators are *right associative* if the tacit parenthesis is around the rightmost operator.

Likewise, $\lceil\, 2 \cdot 3 \cdot 4 \,\rceil^{\cdot}$ could mean $\lceil\, (2 \cdot 3) \cdot 4 \,\rceil^{\cdot}$ or $\lceil\, 2 \cdot (3 \cdot 4) \,\rceil^{\cdot}$. The difference is marginal because $\lceil\, (2 \cdot 3) \cdot 4 \,\rceil^{\cdot}$ and $\lceil\, 2 \cdot (3 \cdot 4) \,\rceil^{\cdot}$ are equal. Nevertheless, Map

---

$\lceil\, \mathsf{x} \overset{\smile}{\rightarrow} \mathsf{y} \,\rceil^{\cdot}$ x associates as y
associativity rule
left associative
right associative

insists that every expression in the text it reads shall have exactly one syntax tree. The arbitrary choice $[\,x \cdot y \cdot z \mathrel{\dot{\to}} (x \cdot y) \cdot z\,]$ makes $[\,\cdot\,]^\circ$ left associative.

See Appendix A.3 for a complete collection of associativity rules.

Exercise 1.14.1 Insert the tacit parentheses in the following expressions, draw their syntax trees and compute their values.

a $[\,1-2-3-4\,]^{\cdot}$

b $[\,2\cdot 3\cdot 4\,]^{\cdot}$

1.15 Equal priority

$[\,x+y\,\boxed{\doteq}\,x-y\,]$ tells Map that $[\,+\,]^\circ$ and $[\,-\,]^\circ$ have the same priority and the same associativity. As an example, $[\,2+3-4\,]^{\cdot}$ means $[\,(2+3)-4\,]^{\cdot}$ because $[\,+\,]^\circ$ and $[\,-\,]^\circ$ have the same priority and are left associative. Likewise, $[\,2-3+4\,]^{\cdot}$ means $[\,(2-3)+4\,]^{\cdot}$.

$[\,x\cdot y \mathrel{\dot{>}} x+y \doteq x-y\,]$ combines two priority statements into one, namely $[\,x\cdot y \mathrel{\dot{>}} x+y\,]$ and $[\,x+y \doteq x-y\,]$. Map accepts priority rules to be stated any number of times but protests against rules that contradict each other. Map would protest against $[\,x+y \mathrel{\dot{>}} x\cdot y\,]^{\cdot}$ if you removed the dot.

Exercise 1.15.1 Insert the tacit parentheses in the following expressions, draw their syntax trees and compute their values.

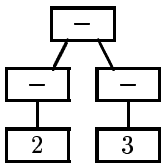
a $[\,1\cdot 2+3\cdot 4+5\cdot 6\,]^{\cdot}$

b $[\,2\cdot 3\cdot(4-5+6)\,]^{\cdot}$

1.16 Unary minus

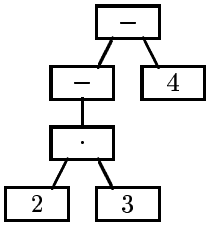
$[\,\boxed{-x}\,]^{\cdot}$ is a unary prefix operator. It changes the sign of $[\,x\,]^{\cdot}$. I will refer to $[\,-x\,]^{\cdot}$ as *unary minus*.

$[\,x\cdot y \mathrel{\dot{>}} -x \mathrel{\dot{>}} x-y\,]$ states that $[\,-x\,]^{\cdot}$ has lower priority than $[\,x\cdot y\,]^{\cdot}$ and higher priority than $[\,x-y\,]^{\cdot}$. As an example, $[\,-2--3\,]^{\cdot}$ means $[\,(-2)-(-3)\,]^{\cdot}$, i.e.



and $[\,-2\cdot 3-4\,]^{\cdot}$ means $[\,(-(2\cdot 3))-4\,]^{\cdot}$, i.e.

$[\,x \doteq y\,]$	x same priority y
$[\,-x\,]^{\cdot}$	minus x
	unary minus



To Map, $[\,-x\,]^{\cdot}$ and $[\,x-y\,]^{\cdot}$ are two different operators with no particular relation to each other. You and I can see that $[\,-x\,]^{\cdot}$ and $[\,x-y\,]^{\cdot}$ are quite similar graphically, but Map is blind to that similarity.

$\left[\boxed{+x}\right]^{\cdot}$ is a (somewhat superfluous) operator that does not change the sign of $[\,x\,]^{\cdot}$. I will refer to $[\,+x\,]^{\cdot}$ as *unary plus*. $[\,+x \mathrel{\breve{=}} -x\,]$ assigns a suitable priority to unary plus.

Exercise 1.16.1 Insert the tacit parentheses in the following expressions, draw their syntax trees and compute their values.

a $[\,----5\,]^{\cdot}$

b $[\,-2\cdot-3\,]^{\cdot}$

c $[\,1---2\,]^{\cdot}$

d $[\,1+-+2\,]^{\cdot}$

1.17 Successor and predecessor

$\left[\,x^{+}\ \boxed{\doteq}\ x+1\,\right]$ defines $\left[\boxed{x^{+}}\right]^{\cdot}$ as a unary suffix operator. $[\doteq]^{\circ}$ is exactly like $[\dot{=}]^{\circ}$ except that $[\doteq]^{\circ}$ may affect the time it takes Map to compute the defined concept as I explain in Section 1.19.

$\left[\boxed{x^{-}} \doteq x-1\,\right]^{\cdot}$ defines $[\,x^{-}\,]^{\cdot}$ as another unary suffix operator. $[\,x^{+}\,]^{\cdot}$ and $[\,x^{-}\,]^{\cdot}$ are known as the "successor" and the "predecessor" operations.

Exercise 1.17.1 Compute the following

a $[\,0^{+++}\,]^{\cdot}$

b $[\,0^{---}\,]^{\cdot}$

$[\,+x\,]^{\cdot}$	plus x unary plus
$[\,x \doteq y\,]^{\cdot}$	x optimised equal y
$[\,x^{+}\,]^{\cdot}$	x successor
$[\,x^{-}\,]^{\cdot}$	x predecessor

1.18 Overview of equal signs

You have now seen the equal signs $[\,\ddot{=}\,]^\circ$, $[\,\hat{=}\,]^\circ$ and $[\,\breve{=}\,]^\circ$ in addition to $[\,\equiv\,]^\circ$, $[\,\doteq\,]^\circ$ and $[\,=\,]^\circ$. Here is an overview of what they mean:

$[\,\mathcal{A} \equiv \mathcal{B}\,]$ means that $[\,\mathcal{A}\,]$ equals $[\,\mathcal{B}\,]$.
$[\,\mathcal{A} \doteq \mathcal{B}\,]$ means that $[\,\mathcal{A}\,]$ is defined to be equal to $[\,\mathcal{B}\,]$.
$[\,\mathcal{A} = \mathcal{B}\,]$ asks Map to check whether or not $[\,\mathcal{A}\,]$ equals $[\,\mathcal{B}\,]$.
$[\,\mathcal{A} \ddot{=} \mathcal{B}\,]$ means that $[\,\mathcal{A}\,]$ is shorthand for $[\,\mathcal{B}\,]$.
$[\,\mathcal{A} \breve{=} \mathcal{B}\,]$ means that $[\,\mathcal{A}\,]$ and $[\,\mathcal{B}\,]$ have the same priority.
$[\,\mathcal{A} \hat{=} \mathcal{B}\,]$ formally means the same as $[\,\mathcal{A} \doteq \mathcal{B}\,]$.

See Section 1.19 for further explanation of $[\,\mathcal{A} \hat{=} \mathcal{B}\,]$. See Section 2.11, Section 2.12, Section 2.14, and Section 3.4 for examples of differences between $[\,\mathcal{A} \equiv \mathcal{B}\,]$ and $[\,\mathcal{A} = \mathcal{B}\,]$.

1.19 Optimising definitions

$[\,\mathcal{A} \hat{=} \mathcal{B}\,]$ formally means the same as $[\,\mathcal{A} \doteq \mathcal{B}\,]$. However, every Map (i.e. every version of the Map software system) has a finite number of constructs that it can compute extremely efficiently using the hardware of the underlying computer. I call such constructs 'hard-coded'.

Some Map's have many hard-coded constructs, some have few. Programs tend to run faster on Map's with many hard-coded constructs. Map's with few hard-coded constructs are toys that can merely be used for mathematical exploration. Map's with many hard-coded constructs can be used for developing industrial quality software.

For each Map, every hard-coded construct has an 'official definition'. Whenever such a Map sees a construct $[\,\mathcal{A}\,]$ defined by a statement of form $[\,\mathcal{A} \hat{=} \mathcal{B}\,]$, it scans through its list of official definitions. If it finds $[\,\mathcal{B}\,]$ in the list, then Map uses its hardware version of the construct instead of the official definition. If Map does not find $[\,\mathcal{B}\,]$ in its list of official definitions, then it treats $[\,\hat{=}\,]^\circ$ as $[\,\doteq\,]^\circ$. This ensures that all Map's can compute all functions, but clever Map's can do so faster.

As an example, when Map sees $[\,\mathsf{x}^+ \hat{=} \mathsf{x} + 1\,]$, it scans through its list of official definitions. If it finds $[\,\mathsf{x} + 1\,]$ in the list, then $[\,\mathsf{x}^+\,]$ will be computed by whatever means Map has for adding one to a number using the underlying computer hardware. If Map does not find $[\,\mathsf{x} + 1\,]$ in its list, then Map will compute $[\,\mathsf{x}^+\,]$ as $[\,\mathsf{x} + 1\,]$ using $[\,+\,]^\circ$.

In special cases, use of $[\,\hat{=}\,]^\circ$ instead of $[\,\doteq\,]^\circ$ may lead to severe decreases in performance (i.e. Map uses more time to compute constructs). You should use $[\,\doteq\,]^\circ$ except when you really expect Map to provide a hard-coded function.

1.20 Terms and statements

The expression $[\,2+3+4\,]^{\cdot}$ has a value, namely $[\,9\,]^{\cdot}$. $[\,2+3+\mathsf{x}\,]^{\cdot}$ has a value for each value of $[\,\mathsf{x}\,]^{\cdot}$. I will refer to expressions that have a value as a *term* so $[\,2+3+4\,]^{\cdot}$ and $[\,2+3+\mathsf{x}\,]^{\cdot}$ are terms.

The expression $[\,f_3(\mathsf{x}) \doteq 2\cdot\mathsf{x}+4\,]^{\cdot}$ has no value. The expression tells Map that $[\,f_3(\mathsf{x})\,]^{\cdot}$ is an operator and it tells Map how to compute $[\,f_3(\mathsf{x})\,]^{\cdot}$. I will refer to expressions that have no value as a *statement*.

I refer to constructs that form terms as *operators* (c.f. Section 1.8). Here you have some operators: $[\,\mathsf{x}+\mathsf{y}\,]^{\cdot}$, $[\,\mathsf{x}\cdot\mathsf{y}\,]^{\cdot}$, and $[\,-\mathsf{x}\,]^{\cdot}$. Variables and numbers are also operators. Furthermore, constructs introduced with $[\doteq]^{\circ}$ and $[\doteqdot]^{\circ}$ are operators. As examples, $[\,f_3(\mathsf{x})\,]^{\cdot}$ and $[\,\mathsf{x}^{+}\,]^{\cdot}$ are operators.

An expression is a term if all constructs in it are operators.

I refer to constructs that form statements as *directives*. Here you have some directives: $[\,\mathsf{x}\doteq\mathsf{y}\,]^{\cdot}$, $[\,\mathsf{x}\doteqdot\mathsf{y}\,]^{\cdot}$, $[\,\mathsf{x}\doteqdot\mathsf{y}\,]^{\cdot}$, $[\,\mathsf{x}\doteqdot\mathsf{y}\,]^{\cdot}$, and $[\,\mathsf{x}\dot{>}\mathsf{y}\,]^{\cdot}$.

In general, Map knows three kinds of constructs: operators, directives, and macros.

In this book, $[\,\mathsf{x}=\mathsf{y}\,]^{\cdot}$ is an operator and $[\,\mathsf{x}\equiv\mathsf{y}\,]^{\cdot}$ is a directive. Until further, just note this as a peculiarity of mine. See Section 2.12 and Section 17.15 for more information.

You can look up constructs like $[\,\mathsf{x}+\mathsf{y}\,]^{\cdot}$ and $[\,\mathsf{x}\doteq\mathsf{y}\,]^{\cdot}$ in the reference manual in Appendix A. In Appendix A you can, among other, see whether a construct is an operator, a directive, or a macro.

1.21 Notational freedom

I have now talked a lot about 'syntax'. The syntax of an expression is the shape and structure of the expression. As an example, $[\,2+3\,]^{\cdot}$ is a syntactic representation of the sum of $[\,2\,]^{\cdot}$ and $[\,3\,]^{\cdot}$.

There is 'notational freedom' in the mathematical world. Notational freedom means that each author can choose any notation he or she likes. The reader of a math book must accept the notation in it or stop reading the book—take it or leave it. I have chosen to denote the sum of two and three by $[\,2+3\,]^{\cdot}$, and you have to tolerate that because otherwise you would restrict my notational freedom. You are free to use any notation you like for the sum of $[\,2\,]^{\cdot}$ and $[\,3\,]^{\cdot}$ next time you write mathematics, but remember to explain your notation if you cannot expect your readers to understand it right away.

Preferably, notation should be consistent within each piece of mathematical work, but an author may use one notational system in one piece of work and another in another.

---

term
statement
operator
directive

Mathematics is not tied to any particular alphabet, language, nationality, race, colour, religion or sex. The author of a book uses his or her mathematical freedom to choose alphabets and languages for that particular book, but that does not tie mathematics in general to those alphabets and languages.

I have chosen English as main language in this book and I have chosen the Latin and Greek alphabets as main alphabets. However, I use $[\![\, \mathbf{Z} \,]\!]$ to denote the set of integers (German $[\![$ Zahlen $]\!]^\circ$), I use Arabic numbers like $[\![\, 2 \,]\!]$ and $[\![\, 10 \,]\!]$, and I use the Hebraic letter $[\![\, \aleph \,]\!]$ (aleph) to denote sizes of sets. I have measured relative to the Christian zero of time in the copyright notice. I have used my notational freedom to make these choices, but the choices are inessential to the mathematical contents of this book and do not restrict anybody who wants to use this mathematics.

When I designed Map, I tried to give as much notational freedom to its users as I could. Map is not in any way biased towards a particular alphabet or language and the notation is defined by its users through definitions like $[\![\, f_3(\mathsf{x}) \doteq 2 \cdot \mathsf{x} + 4 \,]\!]$. Alphabets can be arbitrarily large and Map assigns no meaning to any character in advance. Hence, even characters like $[\![\doteq]\!]^\circ$ and square brackets have no special meaning to Map until such a meaning is assigned by the user (I will explain the details later). If Map finds an error in a text, then it complains, and the complaint is formulated in a language defined by the user. That language could be English or French or anything else.

1.22 Answers

Answer 1.4.1

(a) "x power y plus two end".

(b) "x power y end plus two".

(c) "x power two end plus three".

(d) "a power capital a end plus script a".

Answer 1.5.1 $[\![\, 1 + 2 + 3 + 4 \equiv 3 + 3 + 4 \equiv 6 + 4 \equiv 10 \,]\!]$. The computation requires three steps.

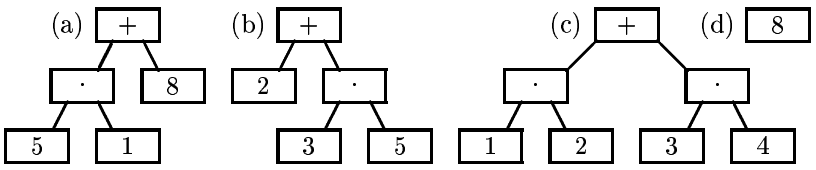
Answer 1.7.1

(a) $[\![\, 4 + (5 \cdot 6) \equiv 4 + 30 \equiv 34 \,]\!]$.

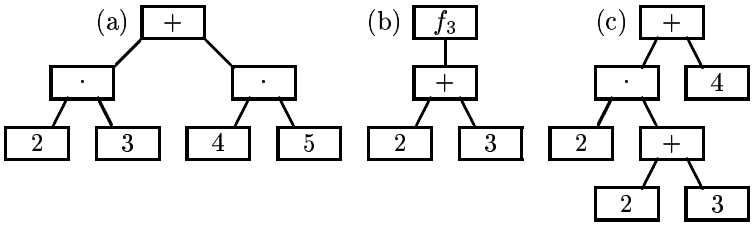
(b) $[\![\, (2 \cdot 3) + (4 \cdot 5) \equiv 6 + (4 \cdot 5) \equiv 6 + 20 \equiv 26 \,]\!]$.

(c) $[\![\, (2 + 3) \cdot 4 \equiv 5 \cdot 4 \equiv 20 \,]\!]$. There are no tacit parentheses.

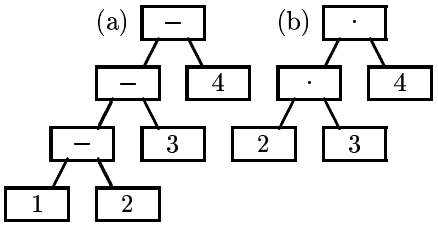
Answer 1.10.1 Values (a) $[\![\, 13 \,]\!]$, (b) $[\![\, 17 \,]\!]$, (c) $[\![\, 14 \,]\!]$, and (d) $[\![\, 8 \,]\!]$.



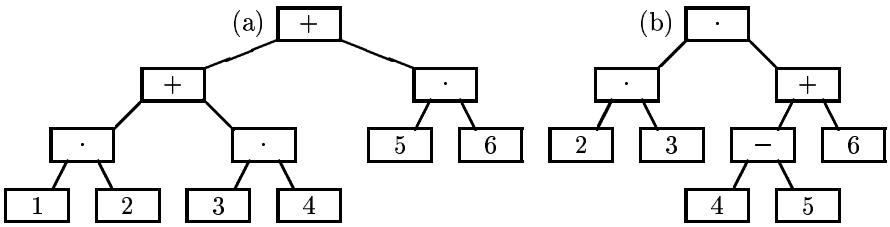
Answer 1.13.1 (a) $[\![\ 26\ ]\!]$, (b) $[\![\ 14\ ]\!]$, and (c) $[\![\ 14\ ]\!]$.



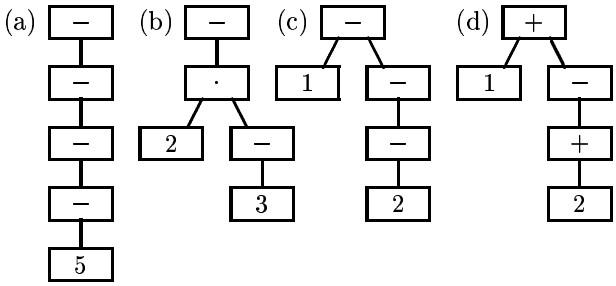
Answer 1.14.1 (a) $[\![\ ((1-2)-3)-4 \equiv -8\ ]\!]$ and (b) $[\![\ (2\cdot 3)\cdot 4 \equiv 24\ ]\!]$



Answer 1.15.1 (a) $[\![\ ((1\cdot 2)+(3\cdot 4))+(5\cdot 6) \equiv 44\ ]\!]$, and (b) $[\![\ (2\cdot 3)\cdot((4-5)+6) \equiv 30\ ]\!]$.



Answer 1.16.1 (a) $[\![\ -(-(-(-5))) \equiv 5\ ]\!]$, (b) $[\![\ -(2\cdot(-3)) \equiv 6\ ]\!]$, (c) $[\![\ 1-(-(-2)) \equiv -1\ ]\!]$, and (d) $[\![\ 1+(-(+2)) \equiv -1\ ]\!]$.



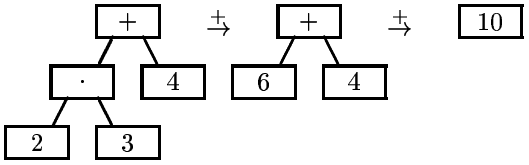
Answer 1.17.1 (a) $[\,3\,]^{\cdot}$, and (b) $[\,-3\,]^{\cdot}$,

Chapter 2

Computation

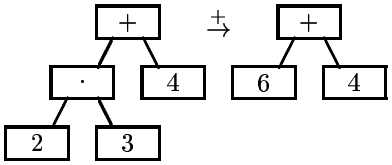
2.1 Reduction

You can express the computation $[\, 2 \cdot 3 + 4 \equiv 6 + 4 \equiv 10 \,]$ by syntax trees:

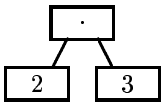


The literature has a lot of names for what is going on in the computation. The computation is a computation by *reduction* or *graph reduction*. The computation falls in two *reduction steps*. A reduction step is a step in a computation where a part of the expression is replaced by something which is equal to that part but "simpler" in some sense.

The first step reads



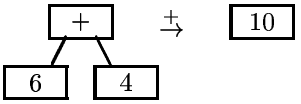
In this step the *redex* is



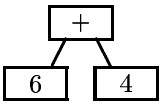
reduction
graph reduction
reduction step
redex

and the *reduct* is

In general, the redex is the part which is going to be replaced and the reduct is what it is replaced by. The second step reads



where the redex is



and the reduct is

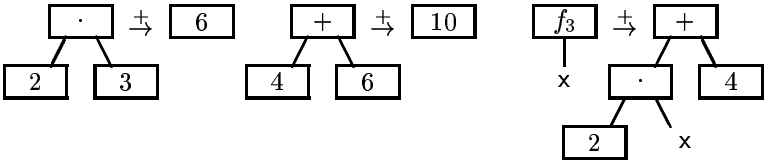
Exercise 2.1.1 Compute $[\,2 \cdot 3 \cdot 4 + 3 \cdot 5\,]^{\cdot}$ by graph reduction.

2.2 Reduction rules

What is "simple" is purely a matter of convention. In the computation $[\,2{\cdot}3{+}4 \equiv 6 + 4 \equiv 10\,]^{\cdot}$, $[\,6\,]^{\cdot}$ is considered simpler than $[\,2 \cdot 3\,]^{\cdot}$ and $[\,10\,]^{\cdot}$ is considered simpler than $[\,6{+}4\,]^{\cdot}$. From a mathematical point of view, however, $[\,2{\cdot}3\,]^{\cdot}$ and $[\,6\,]^{\cdot}$ are just two different syntactic representations of the same number.

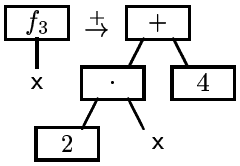
$[\,6\,]^{\cdot}$ is more compact than $[\,2 \cdot 3\,]^{\cdot}$ in the sense that $[\,6\,]^{\cdot}$ is one character where $[\,2 \cdot 3\,]^{\cdot}$ consists of three. However, if you compute $[\,2^{15}\,]^{\cdot}$ then you get $[\,32768\,]^{\cdot}$. $[\,32768\,]^{\cdot}$ is "simpler" than $[\,2^{15}\,]^{\cdot}$, but consists of more characters. Hence, size counted in characters and "simplicity" are not the same. An ancient Roman would probably consider $[\,\mathrm{X}\,]^{\circ}$ simpler than $[\,10\,]^{\cdot}$ and $[\,\mathrm{XIX}\,]^{\circ}$ simpler than $[\,19\,]^{\cdot}$ where a modern Roman would probably have the opposite preference.

It is just a convention that $[\,6\,]^{\cdot}$ is simpler than $[\,2{\cdot}3\,]^{\cdot}$, but it is an important convention. I will write $\left[\,2 \cdot 3 \xrightarrow{\pm} 6\,\right]^{\cdot}$ to state this. In general, a *reduction rule* like $\left[\,\boxed{\mathcal{A} \xrightarrow{\pm} \mathcal{B}}\,\right]^{\cdot}$ both expresses $[\,\mathcal{A} \equiv \mathcal{B}\,]^{\cdot}$ and expresses the convention that $[\,\mathcal{B}\,]^{\cdot}$ is simpler than $[\,\mathcal{A}\,]^{\cdot}$. Here are some examples of reduction rules.

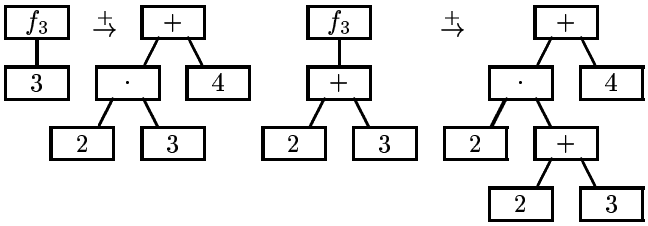


	reduct reduction rule
$[\,\mathrm{x} \xrightarrow{\pm} \mathrm{y}\,]^{\cdot}$	x reduces to y

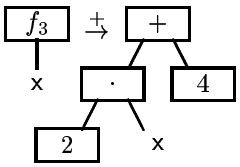
There is no box around $[\![\mathsf{x}]\!]$ in



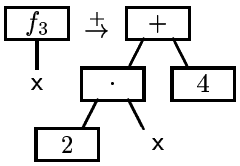
to indicate that $[\![\mathsf{x}]\!]$ can be replaced by any subtree. As examples,



are both instances of



The reduction rule



comes from the definition $[\![f_3(\mathsf{x}) \doteq 2 \cdot \mathsf{x} + 4]\!]$. In general, any definition of form $[\![\mathcal{A} \doteq \mathcal{B}]\!]$ gives rise to a reduction rule $[\![\mathcal{A} \stackrel{\pm}{\to} \mathcal{B}]\!]$. Definitions of form $[\![\mathcal{A} \doteq \mathcal{B}]\!]$ always give rise to reduction rules where the left hand side of $[\![\stackrel{\pm}{\to}]\!]^\circ$ contains exactly one node (the nodes are the boxes, $[\![\mathsf{x}]\!]$ in the reduction rule above is not a node).

When I say $[\![2 \cdot 3 \stackrel{\pm}{\to} 6]\!]$ I actually mean $[\![2 \cdot 3 \equiv 6]\!]$ and that Map computes $[\![2 \cdot 3]\!]$ to $[\![6]\!]$ in one or more steps. Most versions of Map will compute $[\![2 \cdot 3]\!]$ in one step but some will require more. Common for all versions of Map is that computation of $[\![2 \cdot 3]\!]$ yields $[\![6]\!]$ and takes finite, non-zero time.

From $[\![2{\cdot}3{+}4 \stackrel{\pm}{\to} 6{+}4 \stackrel{\pm}{\to} 10]\!]$ you can deduce $[\![2{\cdot}3{+}4 \stackrel{\pm}{\to} 10]\!]$. $[\![2{\cdot}3{+}4 \stackrel{\pm}{\to} 10]\!]$ means that $[\![2 \cdot 3 + 4]\!]$ computes to $[\![10]\!]$ in finite, non-zero time. $[\![10 \stackrel{\pm}{\to} 10]\!]$ does not hold. In general, $[\![10]\!]$ does not reduce to anything else. For that reason I will call $[\![10]\!]$ a *normal form*.

normal form

Exercise 2.2.1 State the reduction rules that correspond to $[\![f_1(x) \doteq x^2 - 1]\!]$ and $\left[\!\left[\boxed{f_5(x,y)} \doteq x^2 - y^2 \right]\!\right]$. Why does $[\![f_4(x) \doteq 2 \cdot x + 4]\!]$ not correspond to a reduction rule?

2.3 Truth, falsehood and selection

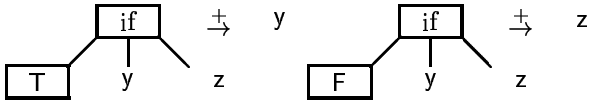
I will use $\left[\!\left[\boxed{T} \right]\!\right]$ to denote truth and $\left[\!\left[\boxed{F} \right]\!\right]$ to denote falsehood. If you want to say that some statement $[\![\mathcal{A}]\!]$ is true, then you should say $[\![\mathcal{A} \equiv T]\!]$. I will refer to $[\![T]\!]$ and $[\![F]\!]$ as *truth values*.

$\left[\!\left[\boxed{if(x,y,z)} \right]\!\right]$ is the most important operation on truth values. I can state what $[\![if(x,y,z)]\!]$ does by the reduction rules

$$[\![if(T,y,z) \xrightarrow{\pm} y]\!], \text{ and}$$

$$[\![if(F,y,z) \xrightarrow{\pm} z]\!].$$

As examples, $[\![if(T,5,6) \xrightarrow{\pm} 5]\!]$ and $[\![if(F,5,6) \xrightarrow{\pm} 6]\!]$. This allows Map to verify $[\![if(T,5,6) = 5]\!]$ and $[\![if(F,5,6) = 6]\!]$. Using syntax trees, the reduction rules read:



Important direct consequences of the reduction rules read

$$[\![if(T,y,z) \equiv y]\!], \text{ and}$$

$$[\![if(F,y,z) \equiv z]\!].$$

The definition $\left[\!\left[\boxed{x\left\{ \begin{matrix} y \\ z \end{matrix} \right.} \doteq if(x,y,z) \right]\!\right]$ introduces $\left[\!\left[x\left\{ \begin{matrix} y \\ z \end{matrix} \right. \right]\!\right]$ as shorthand for $[\![if(x,y,z)]\!]$. Hence, you have

$$\left[\!\left[T\left\{ \begin{matrix} y \\ z \end{matrix} \right. \equiv y \right]\!\right], \text{ and}$$

$[\![f_5(x,y)]\!]$	f five of x comma y end
$[\![T]\!]$	true
$[\![F]\!]$	false
	truth value
$[\![if(x,y,z)]\!]$	if x then y else z end
$\left[\!\left[x\left\{ \begin{matrix} y \\ z \end{matrix} \right. \right]\!\right]$	x select y else z end

$$\left[\mathsf{F}\left\{\begin{array}{l}\mathsf{y}\\ \mathsf{z}\end{array}\equiv\mathsf{z}\right]^{\cdot}.$$

Here you have some examples:

$$\left[\mathsf{T}\left\{\begin{array}{l}5\\ 6\end{array}\equiv 5\right]^{\cdot}.$$

$$\left[\mathsf{F}\left\{\begin{array}{l}5\\ 6\end{array}\equiv 6\right]^{\cdot}.$$

Exercise 2.3.1 I define

$$\left[\mathsf{x}\dot{\wedge}\mathsf{y}\doteq\mathsf{x}\left\{\begin{array}{l}\mathsf{y}\\ \mathsf{F}\end{array}\right]^{\cdot}.$$

You compute $[\,\mathsf{T}\dot{\wedge}\mathsf{T}\,]^{\cdot}$, $[\,\mathsf{T}\dot{\wedge}\mathsf{F}\,]^{\cdot}$, $[\,\mathsf{F}\dot{\wedge}\mathsf{T}\,]^{\cdot}$ and $[\,\mathsf{F}\dot{\wedge}\mathsf{F}\,]^{\cdot}$.

2.4 Numbers

I have already used numbers like $[\,2\,]^{\cdot}$ and $[\,5\,]^{\cdot}$ in formulas. Later, I will give a comprehensive treatment of numbers, their definition and behaviour.

I will refer to whole numbers like $[\,0\,]^{\cdot}$, $[\,7\,]^{\cdot}$, and $[\,-8\,]^{\cdot}$ as *integers*. Furthermore, I will refer to non-negative integers as *natural numbers*. Peano (Italian mathematician, 1858–1932) and Dedekind (German mathematician, 1831–1916) both published fundamental results concerning natural numbers. These mathematicians and many others do not regard $[\,0\,]^{\cdot}$ as a natural number. However, the tendency today is to count $[\,0\,]^{\cdot}$ among the natural numbers, and I will follow that tendency. Hence, in this book, the natural numbers are $[\,0\,]^{\cdot}$, $[\,1\,]^{\cdot}$, $[\,2\,]^{\cdot}$ and so on whereas the positive integers are $[\,1\,]^{\cdot}$, $[\,2\,]^{\cdot}$ and so on.

$[\,3.14159\,]^{\cdot}$ is an example of a number that is not an integer.

2.5 Predicates

If $[\,\mathsf{x}\,]^{\cdot}$ and $[\,\mathsf{y}\,]^{\cdot}$ are numbers, then $\left[\boxed{\mathsf{x}=\mathsf{y}}\right]^{\cdot}$ equals $[\,\mathsf{T}\,]^{\cdot}$ if $[\,\mathsf{x}\,]^{\cdot}$ and $[\,\mathsf{y}\,]^{\cdot}$ have the same magnitude and $[\,\mathsf{x}=\mathsf{y}\,]^{\cdot}$ equals $[\,\mathsf{F}\,]^{\cdot}$ otherwise. Likewise, you have that $\left[\boxed{\mathsf{x}<\mathsf{y}}\right]^{\cdot}$ equals $[\,\mathsf{T}\,]^{\cdot}$ if $[\,\mathsf{x}\,]^{\cdot}$ is less than $[\,\mathsf{y}\,]^{\cdot}$ and that $[\,\mathsf{x}<\mathsf{y}\,]^{\cdot}$ equals $[\,\mathsf{F}\,]^{\cdot}$ otherwise. As examples you have

$$[\,2=2\equiv\mathsf{T}\,]^{\cdot},$$

	integer
	natural number
$[\,\mathsf{x}=\mathsf{y}\,]^{\cdot}$	x computational equal y
$[\,\mathsf{x}<\mathsf{y}\,]^{\cdot}$	x strongly less than y

$[\![2 = 3 \equiv \mathsf{F}]\!]$,

$[\![2 < 3 \equiv \mathsf{T}]\!]$,

$[\![3 < 2 \equiv \mathsf{F}]\!]$, and

$[\![2 < 2 \equiv \mathsf{F}]\!]$.

Note that $[\![2 = 2 \equiv \mathsf{T}]\!]$ means $[\![(2 = 2) \equiv \mathsf{T}]\!]$ according to the priority rules in Section A.2.

When Map reads $[\![f_1(10) = 99]\!]$ then Map computes the value of $[\![f_1(10) = 99]\!]$ as follows:

$$[\![f_1(10) = 99 \stackrel{+}{\rightarrow} 10^2 - 1 = 99 \stackrel{+}{\rightarrow} 100 - 1 = 99 \stackrel{+}{\rightarrow} 99 = 99 \stackrel{+}{\rightarrow} \mathsf{T}]\!].$$

In general, when Map sees a computable expression like $[\![f_1(10) = 99]\!]$, then it computes the expression and accepts it if it computes to $[\![\mathsf{T}]\!]$. Otherwise, Map protests against the expression. As an example, Map would protest against $[\![f_1(10) = 98]\!]$ if the dot were removed since $[\![f_1(10) = 98]\!]$ computes to $[\![\mathsf{F}]\!]$. Map would also protest against $[\![f_1(10)]\!]$ since $[\![f_1(10)]\!]$ computes to $[\![99]\!]$ which differs from $[\![\mathsf{T}]\!]$.

The predicates $[\![\boxed{\mathsf{x} \neq \mathsf{y}}]\!]$, $[\![\boxed{\mathsf{x} \leq \mathsf{y}}]\!]$, $[\![\boxed{\mathsf{x} > \mathsf{y}}]\!]$, and $[\![\boxed{\mathsf{x} \geq \mathsf{y}}]\!]$ have the obvious meaning. Here are some examples: $[\![2 \neq 2 \equiv \mathsf{F}]\!]$, $[\![2 \leq 3 \equiv \mathsf{T}]\!]$ and $[\![2 \geq 3 \equiv \mathsf{F}]\!]$.

Exercise 2.5.1 Draw syntax trees of (a) $[\![8 < -11]\!]$ and (b) $[\![\mathrm{if}(3 < 4, 5, 6)]\!]$ and compute their values.

2.6 Recursive definitions

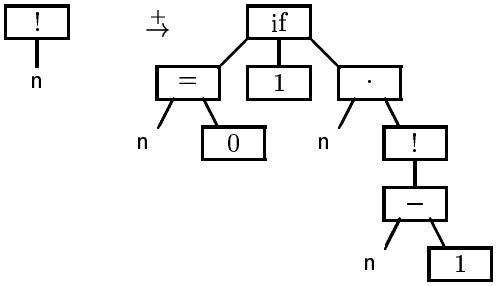
$[\![\boxed{\mathsf{n!}} \doteq \mathrm{if}(\mathsf{n} = 0, 1, \mathsf{n} \cdot (\mathsf{n} - 1)!)]\!]$ is a *recursive* definition. Recursive definitions are *circular* in the sense that the defined concept $[\![\mathsf{x}!]\!]$ occurs within its own definition.

The priority rule $[\![\mathsf{x}! \stackrel{\smile}{>} \mathsf{x} \cdot \mathsf{y}]\!]$ says that the definition means $[\![\mathsf{n}! \doteq \mathrm{if}(\mathsf{n} = 0, 1, \mathsf{n} \cdot ((\mathsf{n} - 1)!))]\!]$. Another way to state the same definition is:

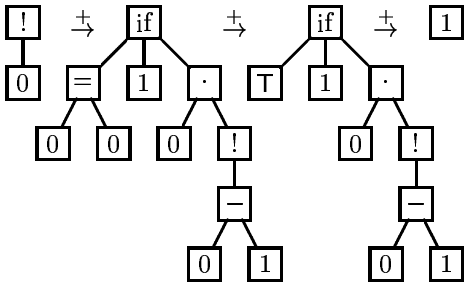
$$\left[\!\!\left[\mathsf{n}! \doteq \mathsf{n} = 0 \left\{ \begin{array}{l} 1 \\ \mathsf{n} \cdot (\mathsf{n} - 1)! \end{array} \right. \right]\!\!\right].$$

The definition gives rise to the following reduction rule:

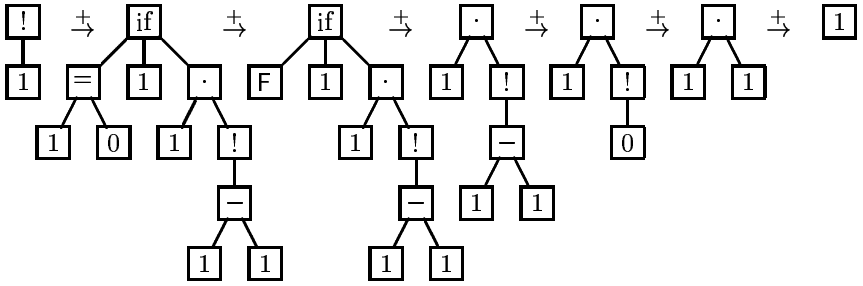
$[\![\mathsf{x} \neq \mathsf{y}]\!]$	x computational unequal y
$[\![\mathsf{x} \leq \mathsf{y}]\!]$	x weakly less than y
$[\![\mathsf{x} > \mathsf{y}]\!]$	x strongly greater than y
$[\![\mathsf{x} \geq \mathsf{y}]\!]$	x weakly greater than y
$[\![\mathsf{x}!]\!]$	x faculty
	recursive
	circular



Map can compute $[\![\mathsf{n}!]\!]^{\cdot}$ for all natural numbers $[\![\mathsf{n}]\!]^{\cdot}$ using this reduction rule. Map computes $[\![0!]\!]^{\cdot}$ to $[\![1]\!]^{\cdot}$ as follows:



Map computes $[\![1!]\!]^{\cdot}$ to $[\![1]\!]^{\cdot}$ as follows.



You have

$[\![0! = 1]\!]$,

$[\![1! = 1]\!]$,

$[\![2! = 2 \cdot 1! = 2 \cdot 1]\!]$,

$[\![3! = 3 \cdot 2! = 3 \cdot 2 \cdot 1]\!]$, and

$[\![4! = 4 \cdot 3! = 4 \cdot 3 \cdot 2 \cdot 1]\!]$.

In general you have

$$[\![\mathsf{n}! = \mathsf{n} \cdot (\mathsf{n} - 1) \cdot (\mathsf{n} - 2) \cdots 3 \cdot 2 \cdot 1]\!]^{\circ}.$$

The literature calls $[\![\, \mathsf{n}! \,]\!]$ the *faculty function*.

As you can see, Map can easily use the definition $[\![\, \mathsf{n}! \doteq \mathrm{if}(\mathsf{n} = 0, 1, \mathsf{n}\cdot(\mathsf{n}-1)!) \,]\!]$ even though the definition is fishy: The definition of faculty refers to faculty itself, so the definition is circular. If you say "left is defined as the opposite of right and right is defined as the opposite of left", then you have another circular definition. This definition of left and right is not very helpful to someone who doesn't know what left and right is. On the contrary, the definition of faculty is helpful in the sense that it allows Map to compute the faculty function.

Exercise 2.6.1 I define $[\![\, f_6(\mathsf{x}) \doteq \mathrm{if}(\mathsf{x} = 0, 0, f_6(\mathsf{x}-1)+2) \,]\!]$.

(a) Compute $[\![\, f_6(0) \,]\!]$.

(b) Compute $[\![\, f_6(1) \,]\!]$.

(c) Compute $[\![\, f_6(2) \,]\!]$.

(d) Compute $[\![\, f_6(3) \,]\!]$.

(e) What does $[\![\, f_6(\mathsf{n}) \,]\!]$ compute for natural numbers $[\![\, \mathsf{n} \,]\!]$?

(f) Compute $[\![\, f_6(-1) \,]\!]$

2.7 Bottom $[\![\, \bot \,]\!]$

If $[\![\, \mathsf{n} \neq 0 \,]\!]$ then you have the reduction rule $\left[\!\left[\, \mathsf{n}! \xrightarrow{+} \mathsf{n}\cdot(\mathsf{n}-1)! \,\right]\!\right]$. Hence, Map will compute $[\![\, (-1)! \,]\!]$ by

$$\begin{array}{lcl} [\![\, (-1)! & \xrightarrow{+} & (-1)\cdot(-2)! \\ & \xrightarrow{+} & (-1)\cdot(-2)\cdot(-3)! \\ & \xrightarrow{+} & (-1)\cdot(-2)\cdot(-3)\cdot(-4)! \\ & \xrightarrow{+} & \cdots \,]\!]^{\circ} \end{array}$$

As you can see, the computation never ends and yields no information about the value of $[\![\, (-1)! \,]\!]$.

If computation of a term yields no information about the value of the term, then I will call the term *non-informative*. Otherwise, I will call the term *informative*. As an example, $[\![\, (-1)! \,]\!]$ is non-informative. Computation of $[\![\, (-2)! \,]\!]$ does not yield any information either, so it is non-informative as well. $[\![\, 2+3 \,]\!]$ is informative because computation yields $[\![\, 5 \,]\!]$ in finite time.

I make the convention that if two terms are non-informative, then they are equal to each other. As an example,

$$[\![\, (-1)! \equiv (-2)! \,]\!].$$

---

faculty function
non-informative
informative

I make the convention that a non-informative term differs from any informative term:

$$\lceil\!\lceil\, (-1)! \not\equiv 2 + 3 \,\rceil\!\rceil, \text{ and}$$

$$\lceil\!\lceil\, (-2)! \not\equiv 2 + 3 \,\rceil\!\rceil.$$

Two informative terms may of course differ:

$$\lceil\!\lceil\, 2 + 3 \not\equiv 2 + 4 \,\rceil\!\rceil.$$

I follow the literature and use $\lceil\!\lceil\, \boxed{\bot} \,\rceil\!\rceil$ to denote the value of non-informative terms. Here you have some examples:

$$\lceil\!\lceil\, (-1)! \equiv \bot \,\rceil\!\rceil,$$

$$\lceil\!\lceil\, (-2)! \equiv \bot \,\rceil\!\rceil, \text{ and}$$

$$\lceil\!\lceil\, 2 + 3 \not\equiv \bot \,\rceil\!\rceil.$$

As you will see in Section 17.7, there is a hierarchy of more and less informative values with $\lceil\!\lceil\, \bot \,\rceil\!\rceil$ at the bottom. For that reason, $\lceil\!\lceil\, \bot \,\rceil\!\rceil$ reads "bottom".

$\lceil\!\lceil\, \bot \,\rceil\!\rceil$ is radically different from other mathematical and computational concepts like $\lceil\!\lceil\, \mathsf{T} \,\rceil\!\rceil$ and $\lceil\!\lceil\, \mathsf{F} \,\rceil\!\rceil$ and $\lceil\!\lceil\, 2 \,\rceil\!\rceil$ and $\lceil\!\lceil\, 5 \,\rceil\!\rceil$. $\lceil\!\lceil\, \mathsf{T} \,\rceil\!\rceil$ and $\lceil\!\lceil\, \mathsf{F} \,\rceil\!\rceil$ and $\lceil\!\lceil\, 2 \,\rceil\!\rceil$ and $\lceil\!\lceil\, 5 \,\rceil\!\rceil$ all contain information, whereas $\lceil\!\lceil\, \bot \,\rceil\!\rceil$ denotes absence of information. $\lceil\!\lceil\, \bot \,\rceil\!\rceil$ will never occur in a computation.

2.8 Strictness of $\lceil\!\lceil\, x + y \,\rceil\!\rceil$

Computation of $\lceil\!\lceil\, (-1)! + 1 \,\rceil\!\rceil$ proceeds thus:

$$\begin{aligned} \lceil\!\lceil\, (-1)! + 1 &\overset{+}{\to} (-1) \cdot (-2)! + 1 \\ &\overset{+}{\to} (-1) \cdot (-2) \cdot (-3)! + 1 \\ &\overset{+}{\to} (-1) \cdot (-2) \cdot (-3) \cdot (-4)! + 1 \\ &\overset{+}{\to} \cdots \,\rceil\!\rceil^{\circ} \end{aligned}$$

Hence,

$$\lceil\!\lceil\, (-1)! + 1 \equiv \bot \,\rceil\!\rceil.$$

From the previous section you have $\lceil\!\lceil\, (-1)! \equiv \bot \,\rceil\!\rceil$. If you replace $\lceil\!\lceil\, (-1)! \,\rceil\!\rceil$ by $\lceil\!\lceil\, \bot \,\rceil\!\rceil$ in $\lceil\!\lceil\, (-1)! + 1 \equiv \bot \,\rceil\!\rceil$ you get

$$\lceil\!\lceil\, \bot + 1 \equiv \bot \,\rceil\!\rceil.$$

As you can see, my convention that $\lceil\!\lceil\, \bot \,\rceil\!\rceil$ equals any non-informative term is enough to deduce $\lceil\!\lceil\, \bot + 1 \equiv \bot \,\rceil\!\rceil$.

$\lceil\!\lceil\, \bot \,\rceil\!\rceil$ bottom

Here you have some rules about $[\![\ \bot\ ]\!]$:

$[\![\ \bot + y \equiv \bot\ ]\!]$, and

$[\![\ x + \bot \equiv \bot\ ]\!]$.

I will say that $[\![\ x+y\ ]\!]$ is *strict in* $[\![\ x\ ]\!]$ because $[\![\ x+y \equiv \bot\ ]\!]$ if $[\![\ x \equiv \bot\ ]\!]$ regardless of the value of $[\![\ y\ ]\!]$. $[\![\ x + y\ ]\!]$ is also strict in $[\![\ y\ ]\!]$. Furthermore I will say that $[\![\ x+y\ ]\!]$ is *strict* because it is strict in all of its variables. If you look up $[\![\ x+y\ ]\!]$ in the reference manual (Section A.80), then you will find the rules $[\![\ \bot+x \equiv \bot\ ]\!]$ and $[\![\ x + \bot \equiv \bot\ ]\!]$.

2.9 Strictness of $[\![\ \mathrm{if}(a, b, c)\ ]\!]$

When Map computes $[\![\ \mathrm{if}(a, b, c)\ ]\!]$ it first computes $[\![\ a\ ]\!]$. If $[\![\ a\ ]\!]$ is $[\![\ T\ ]\!]$ then Map reduces $[\![\ \mathrm{if}(a, b, c)\ ]\!]$ to $[\![\ b\ ]\!]$. If $[\![\ a\ ]\!]$ is $[\![\ F\ ]\!]$ then Map reduces $[\![\ \mathrm{if}(a, b, c)\ ]\!]$ to $[\![\ c\ ]\!]$. However, if computation of $[\![\ a\ ]\!]$ never ends, then Map will never get to a point where it can reduce $[\![\ \mathrm{if}(a, b, c)\ ]\!]$. This shows that if $[\![\ a \equiv \bot\ ]\!]$ then $[\![\ \mathrm{if}(a, b, c) \equiv \bot\ ]\!]$, or

$[\![\ \mathrm{if}(\bot, b, c) \equiv \bot\ ]\!]$.

Hence, $[\![\ \mathrm{if}(a, b, c)\ ]\!]$ is strict in $[\![\ a\ ]\!]$. On the other hand, you have

$[\![\ \mathrm{if}(T, 5, \bot) \equiv 5\ ]\!]$ and

$[\![\ \mathrm{if}(F, \bot, 6) \equiv 6\ ]\!]$

which shows that an expression may differ from $[\![\ \bot\ ]\!]$ even if $[\![\ \bot\ ]\!]$ occurs somewhere in it. I will say that $[\![\ \mathrm{if}(a, b, c)\ ]\!]$ is *non-strict in* $[\![\ b\ ]\!]$ because there are values of $[\![\ a\ ]\!]$ and $[\![\ c\ ]\!]$ for which $[\![\ \mathrm{if}(a, \bot, c)\ ]\!]$ differs from $[\![\ \bot\ ]\!]$. Likewise, $[\![\ \mathrm{if}(a, b, c)\ ]\!]$ is non-strict in $[\![\ c\ ]\!]$.

A construct is *non-strict* if it is non-strict in one or more of its arguments. Therefore, $[\![\ \mathrm{if}(a, b, c)\ ]\!]$ is non-strict.

You could regard $[\![\ \mathrm{if}(\bot, b, c) \xrightarrow{+} \bot\ ]\!]$ as a reduction rule if you regard $[\![\ \bot\ ]\!]$ as "simpler" than $[\![\ \mathrm{if}(\bot, b, c)\ ]\!]$. However, Map cannot execute such reduction rules since Map cannot recognise any computation as never-ending.

It would be very convenient to have a computer program which, given an expression, stated whether or not that expression was informative. However, such a computer program does not exist according to a theorem which is called Turings Halting Theorem.

Exercise 2.9.1 I define $[\![\ x \Rightarrow y \doteq \mathrm{if}(x, y, T)\ ]\!]$. You compute

strict in $[\![\ x\ ]\!]$
strict
non-strict in $[\![\ x\ ]\!]$
non-strict

a $[\,\mathsf{T} \dot{\Rightarrow} \mathsf{T}\,]^{\cdot}$.

b $[\,\mathsf{T} \dot{\Rightarrow} \mathsf{F}\,]^{\cdot}$.

c $[\,\mathsf{T} \dot{\Rightarrow} \bot\,]^{\cdot}$.

d $[\,\mathsf{F} \dot{\Rightarrow} \bot\,]^{\cdot}$.

e $[\,\bot \dot{\Rightarrow} \bot\,]^{\cdot}$.

2.10 Order of computation

Strictness says something about the order of computation. The fact that $[\,\mathrm{if}(\mathsf{a}, \mathsf{b}, \mathsf{c})\,]^{\cdot}$ is strict in $[\,\mathsf{a}\,]^{\cdot}$ and non-strict in $[\,\mathsf{b}\,]^{\cdot}$ and $[\,\mathsf{c}\,]^{\cdot}$ says that Map focuses on $[\,\mathsf{a}\,]^{\cdot}$ first. Map merely computes $[\,\mathsf{b}\,]^{\cdot}$ if $[\,\mathsf{a}\,]^{\cdot}$ is true. Map is *lazy* in the sense that it merely computes $[\,\mathsf{b}\,]^{\cdot}$ if the value of $[\,\mathsf{b}\,]^{\cdot}$ is needed. As you will see later, Map is much more lazy than this (c.f. Section 5.3).

Laziness is a deed in mathematics. Mathematics is constant hard work to find the easy way.

2.11 Strictness of $[\,\mathsf{x} = \mathsf{y}\,]^{\cdot}$

Computation of $[\,(-1)! = 1\,]^{\cdot}$ proceeds thus:

$$\begin{aligned} [\,(-1)! = 1 &\overset{+}{\to} (-1)\cdot(-2)! = 1 \\ &\overset{+}{\to} (-1)\cdot(-2)\cdot(-3)! = 1 \\ &\overset{+}{\to} (-1)\cdot(-2)\cdot(-3)\cdot(-4)! = 1 \\ &\overset{+}{\to} \cdots\,]^{\circ} \end{aligned}$$

Hence,

$$[\,(-1)! = 1 \equiv \bot\,]^{\cdot}.$$

You still have $[\,(-1)! \equiv \bot\,]^{\cdot}$. If you replace $[\,(-1)!\,]^{\cdot}$ by $[\,\bot\,]^{\cdot}$ in $[\,(-1)! = 1 \equiv \bot\,]^{\cdot}$ you get

$$[\,\bot = 1 \equiv \bot\,]^{\cdot}.$$

In general, you have

$$[\,\bot = \mathsf{x} \equiv \bot\,]^{\cdot}, \text{ and}$$

$$[\,\mathsf{x} = \bot \equiv \bot\,]^{\cdot},$$

so $[\,\mathsf{x} = \mathsf{y}\,]^{\cdot}$ is strict. In particular, you have

$$[\,(\bot = \bot) \equiv \bot\,]^{\cdot}.$$

This gives you an example of a difference between $[\,\mathsf{x} = \mathsf{y}\,]^{\cdot}$ and $[\,\mathsf{x} \equiv \mathsf{y}\,]^{\cdot}$:

lazy

- $[\![\bot \equiv \bot]\!]$ holds.
- $[\![\bot = \bot]\!]$ is non-informative.

2.12 $[\![x \equiv y]\!]$ is a directive

$[\![x = y]\!]$ may produce values such as $[\![T]\!]$ and $[\![F]\!]$, so $[\![x = y]\!]$ is clearly an operator.

In this book I have chosen to make $[\![x \equiv y]\!]$ a statement as I mentioned in Section 1.20. As a consequence, $[\![x \equiv y]\!]$ is not allowed inside terms. As an example,

$$\left[\!\!\left[n! \doteq n \equiv 0 \left\{ \begin{array}{l} 1 \\ n \cdot (n-1)! \end{array} \right. \right]\!\!\right]^{\circ}$$

is illegal syntax because $[\![n \equiv 0]\!]$ occurs inside the term to the right of $[\![\doteq]\!]^{\circ}$. The definition of the faculty function reads

$$\left[\!\!\left[n! \doteq n = 0 \left\{ \begin{array}{l} 1 \\ n \cdot (n-1)! \end{array} \right. \right]\!\!\right]$$

which is ok since $[\![=]\!]^{\circ}$ is an operator.

I will refer to statements of form $[\![\mathcal{A} \equiv \mathcal{B}]\!]$ as *equations*. I will avoid saying that an equation is "true" or "false" because that could be taken to mean that the equation had value $[\![T]\!]$ or $[\![F]\!]$. Instead, I will use the words *hold* and *fail*. As an example, I will say that

$$[\![\mathrm{if}(T, x, y) \equiv x]\!]$$

holds because $[\![\mathrm{if}(T, x, y)]\!]$ equals $[\![x]\!]$ for all values of $[\![x]\!]$ and $[\![y]\!]$. I will say that

$$[\![n! \equiv n \cdot (n-1)!]\!]$$

fails because $[\![n!]\!]$ is unequal to $[\![n \cdot (n-1)!]\!]$ for some $[\![n]\!]$.

Exercise 2.12.1 Give a *counterexample* to $[\![n! \equiv n \cdot (n-1)!]\!]$, i.e. find a value for $[\![n]\!]$ that makes the value of $[\![n!]\!]$ unequal to the value of $[\![n \cdot (n-1)!]\!]$.

---

equation
hold
fail
counterexample

2.13 Division

For all integers $[\![a]\!]$ and for all positive integers $[\![b]\!]$ there exists exactly one integer $[\![q]\!]$ and one integer $[\![r]\!]$ such that

$$[\![a \equiv q \cdot b + r]\!] \text{ and}$$

$$[\![0 \leq r < b]\!].$$

$[\![q]\!]$ is the integer quotient when dividing $[\![a]\!]$ by $[\![b]\!]$, and $[\![r]\!]$ is the remainder. I will not prove the existence and uniqueness of $[\![q]\!]$ and $[\![r]\!]$ here. (See [4] for a proof).

I use $\left[\!\left[\boxed{a \mathbin{//} b} \right]\!\right]$ and $\left[\!\left[\boxed{a \mathbin{\%} b} \right]\!\right]$ to denote the uniquely defined $[\![q]\!]$ and $[\![r]\!]$, respectively. Hence, you have

$$[\![a \equiv (a \mathbin{//} b) \cdot b + (a \mathbin{\%} b)]\!] \text{ and}$$

$$[\![0 \leq a \mathbin{\%} b < b]\!].$$

Here you have some examples:

$$[\![182 \mathbin{//} 10 \equiv 18]\!],$$

$$[\![182 \mathbin{\%} 10 \equiv 2]\!],$$

$$[\![(-3) \mathbin{//} 10 \equiv -1]\!], \text{ and}$$

$$[\![(-3) \mathbin{\%} 10 \equiv 7]\!].$$

The last example says that $[\![-3]\!]$ divided by $[\![10]\!]$ gives $[\![-1]\!]$ and a remainder of $[\![7]\!]$. This is correct since

$$[\![-3 \equiv (-1) \cdot 10 + 7]\!] \text{ and}$$

$$[\![0 \leq 7 < 10]\!].$$

I give $[\![\mathbin{//}]\!]^\circ$ and $[\![\mathbin{\%}]\!]^\circ$ the same priority as $[\![\cdot]\!]^\circ$: $[\![x \mathbin{//} y \doteq x \mathbin{\%} y \doteq x \cdot y]\!]$. $[\![\mathbin{//}]\!]^\circ$ and $[\![\mathbin{\%}]\!]^\circ$ are strict.

Exercise 2.13.1 Compute the following:

(a) $[\![7 \mathbin{//} 3]\!]$.

(b) $[\![7 \mathbin{\%} 3]\!]$.

(c) $[\![(-7) \mathbin{//} 3]\!]$.

(d) $[\![(-7) \mathbin{\%} 3]\!]$.

2.14 Exceptions

$[\![5]\!]$ divided by $[\![0]\!]$ is undefined. Now, what should Map do if you ask it to compute $[\![5 \mathbin{//} 0]\!]$? I have decided to introduce $\left[\!\left[\boxed{\bullet} \right]\!\right]$ to stand for "exception",

$[\![x \mathbin{//} y]\!]$	x integer divide y
$[\![x \mathbin{\%} y]\!]$	x modulo y
$[\![\bullet]\!]$	exception

and I have decided that $[\![5 \mathbin{//} 0 \equiv \bullet]\!]$ (this follows from the definition of $[\![x \mathbin{//} y]\!]$ which I state later). In general, $[\![a \mathbin{//} b \equiv \bullet]\!]$ and $[\![a \mathbin{\%} b \equiv \bullet]\!]$ whenever $[\![a]\!]$ and $[\![b]\!]$ are numbers, and $[\![b \leq 0]\!]$.

If $[\![x]\!]$ is a number, a truth value, or an exception, then

$$[\![x + \bullet \equiv \bullet]\!], \text{ and}$$

$$[\![\bullet + x \equiv \bullet]\!].$$

Similar rules hold for $[\![x - y]\!]$, $[\![x \cdot y]\!]$, $[\![x \mathbin{//} y]\!]$, $[\![x \mathbin{\%} y]\!]$, $[\![x = y]\!]$, $[\![x \neq y]\!]$, $[\![x < y]\!]$, $[\![x \leq y]\!]$, $[\![x > y]\!]$, and $[\![x \geq y]\!]$.

You may look up individual constructs like $[\![x+y]\!]$ and $[\![x-y]\!]$ in Appendix A to get a summary of how each construct reacts to exceptions and other values.

If $[\![n]\!]$ is a number then you have

$$\begin{array}{llll} [\![n + \mathsf{T} \equiv \bullet]\!] & [\![n = \mathsf{T} \equiv \mathsf{F}]\!] & [\![n \neq \mathsf{T} \equiv \mathsf{T}]\!] \\ [\![n + \mathsf{F} \equiv \bullet]\!] & [\![n = \mathsf{F} \equiv \mathsf{F}]\!] & [\![n \neq \mathsf{F} \equiv \mathsf{T}]\!] \\ [\![\mathsf{T} + n \equiv \bullet]\!] & [\![\mathsf{T} = n \equiv \mathsf{F}]\!] & [\![\mathsf{T} \neq n \equiv \mathsf{T}]\!] \\ [\![\mathsf{F} + n \equiv \bullet]\!] & [\![\mathsf{F} = n \equiv \mathsf{F}]\!] & [\![\mathsf{F} \neq n \equiv \mathsf{T}]\!] \end{array}$$

Rules similar to those of $[\![x + y]\!]$ hold for $[\![x - y]\!]$, $[\![x \cdot y]\!]$, $[\![x \mathbin{//} y]\!]$, $[\![x \mathbin{\%} y]\!]$, $[\![x < y]\!]$, $[\![x \leq y]\!]$, $[\![x > y]\!]$, and $[\![x \geq y]\!]$.

If $[\![n]\!]$ is a number, then

$$[\![\mathrm{if}(n, b, c) \equiv \bullet]\!].$$

Furthermore,

$$[\![\mathrm{if}(\bullet, b, c) \equiv \bullet]\!].$$

$[\![\bullet = \bullet]\!]$ equals $[\![\bullet]\!]$ and, hence, $[\![\bullet = \bullet]\!]$ is not true. In contrast, $[\![\bullet \equiv \bullet]\!]$ holds. This gives yet another difference between $[\![x \equiv y]\!]$ and $[\![x = y]\!]$.

Here you have some examples:

$$[\![\bullet + 5 \equiv \bullet]\!],$$

$$[\![\mathsf{T} + 5 \equiv \bullet]\!],$$

$$[\![\mathsf{T} + \bot \equiv \bot]\!],$$

$$[\![\bullet + \bot \equiv \bot]\!],$$

$$[\![5 < \bullet \equiv \bullet]\!],$$

$$[\![5 = \bullet \equiv \bullet]\!],$$

$$[\![5 \leq \mathsf{T} \equiv \bullet]\!],$$

$$[\![5 = \mathsf{T} \equiv \mathsf{F}]\!],$$

$[\![\bullet = \bot \equiv \bot]\!]$,

$[\![\mathrm{if}(\mathsf{T}, 5, 6) \equiv 5]\!]$,

$[\![\mathrm{if}(\mathsf{F}, 5, 6) \equiv 6]\!]$,

$[\![\mathrm{if}(\bullet, 5, 6) \equiv \bullet]\!]$,

$[\![\mathrm{if}(\bullet, \bot, \bot) \equiv \bullet]\!]$,

$[\![\mathrm{if}(2, 5, 6) \equiv \bullet]\!]$, and

$[\![\mathrm{if}(\bot, 5, 6) \equiv \bot]\!]$.

Here is another example. Define $\left[\!\left[\boxed{f_7(\mathsf{x}, \mathsf{y})} \doteq \mathrm{if}(\mathsf{y} = -1, 0, (\mathsf{x} \mathbin{//} \mathsf{y}) + f_7(\mathsf{x}, \mathsf{y}-1)) \right]\!\right]$. You have $[\![f_7(5,2) \equiv \bullet]\!]$ because computation of $[\![f_7(5,2)]\!]$ leads to computation of $[\![f_7(5,1)]\!]$ which in turn leads to computation of $[\![f_7(5,0)]\!]$ which leads to computation of $[\![5 \mathbin{//} 0]\!]$ which is *exceptional*.

$[\![\mathcal{A} \equiv \bot]\!]$ if it takes infinite time to compute $[\![\mathcal{A}]\!]$. By convention, $[\![\mathcal{A} \equiv \bullet]\!]$ means that it takes finite time to see that $[\![\mathcal{A}]\!]$ is "meaningless" in some sense, e.g. that computation of $[\![\mathcal{A}]\!]$ leads to division by zero or addition of a number and a truth value.

Exercise 2.14.1 Compute the following:

(a) $[\![5 \mathbin{\%} 1 + 5 \mathbin{\%} (-1)]\!]$.

(b) $[\![(-1)! + 7 \mathbin{//} 0]\!]$.

2.15 Exception catching

$\left[\!\left[\boxed{\mathsf{x}?} \right]\!\right]$ is true if $[\![\mathsf{x}]\!]$ is an exception and false if $[\![\mathsf{x}]\!]$ is a number or a truth value. In particular,

$[\![\bullet? \equiv \mathsf{T}]\!]$,

$[\![\mathsf{T}? \equiv \mathsf{F}]\!]$,

$[\![\mathsf{F}? \equiv \mathsf{F}]\!]$,

$[\![\bot? \equiv \bot]\!]$, and

$[\![\mathsf{x}? \equiv \mathsf{F}]\!]$ if $[\![\mathsf{x}]\!]$ is a number

---

$[\![f_7(\mathsf{x}, \mathsf{y})]\!]$	f seven of x comma y end exceptional
$[\![\mathsf{x}?]\!]$	x exceptional

As an example you have $[\![f_7(5,2)?]\!]$ since $[\![f_7(5,2) \equiv \bullet]\!]^{\cdot}$. As another example, define

$$\left[\!\!\left[\text{default}(\mathsf{x},\mathsf{y}) \doteq \mathsf{x}?\left\{ \begin{array}{c} \mathsf{y} \\ \mathsf{x} \end{array} \right. \right]\!\!\right],$$

$[\![\mathsf{x}\ \text{DIV}\ \mathsf{y} \doteq \text{default}(\mathsf{x}\ //\ \mathsf{y}, 0)]\!]$ and

$[\![\mathsf{x}\ \text{MOD}\ \mathsf{y} \doteq \text{default}(\mathsf{x}\ \%\ \mathsf{y}, \mathsf{x})]\!]$.

You have that $[\![\mathsf{x}\ \text{DIV}\ \mathsf{y} \equiv \mathsf{x}\ //\ \mathsf{y}]\!]^{\cdot}$ whenever $[\![\mathsf{x}\ //\ \mathsf{y}]\!]^{\cdot}$ is defined and $[\![\mathsf{x}\ \text{DIV}\ \mathsf{y} \equiv 0]\!]^{\cdot}$ otherwise. $[\![\mathsf{x}\ \text{DIV}\ \mathsf{y}]\!]^{\cdot}$ and $[\![\mathsf{x}\ \text{MOD}\ \mathsf{y}]\!]^{\cdot}$ satisfy

$$[\![\mathsf{x} \equiv (\mathsf{x}\ \text{DIV}\ \mathsf{y}) \cdot \mathsf{y} + (\mathsf{x}\ \text{MOD}\ \mathsf{y})]\!]^{\cdot}$$

for all integers $[\![\mathsf{x}]\!]^{\cdot}$ and $[\![\mathsf{y}]\!]^{\cdot}$, even when $[\![\mathsf{y} \leq 0]\!]^{\cdot}$. Of course you do not have $[\![0 \leq \mathsf{x}\ \text{MOD}\ \mathsf{y} < \mathsf{y}]\!]^{\cdot}$ when $[\![\mathsf{y} \leq 0]\!]^{\cdot}$.

Exercise 2.15.1 Compute the following:

(a) $[\![\text{default}(f_7(5,2), 4)]\!]^{\cdot}$.

(b) $[\![\text{default}((-1)!, 4)]\!]^{\cdot}$.

2.16 Operations on truth values

I use the convention that *truth is greater than falsehood*, so $[\![\mathsf{T} > \mathsf{F}]\!]^{\cdot}$ is true. Here you have a table that shows the value of various operations applied to truth values:

x	y	$=$	$\neq$	$<$	$\leq$	$>$	$\geq$	$+$	$-$	$\cdot$	$//$	$\%$
T	T	T	F	F	T	F	T	T	•	T	•	•
T	F	F	T	F	F	T	T	T	•	F	•	•
F	T	F	T	T	T	F	F	T	•	F	•	•
F	F	T	F	F	T	F	T	F	•	F	•	•

Exercise 2.16.1 What are the values of the following:

(a) $[\![5 \geq 3]\!]^{\cdot}$

(b) $[\![5 \geq 5]\!]^{\cdot}$

(c) $[\![5 \geq 10]\!]^{\cdot}$

(d) $[\![5 \geq \mathsf{T}]\!]^{\cdot}$

(e) $[\![5 \geq \bullet]\!]^{\cdot}$

truth is greater than falsehood

(f) $[\,5 \geq \bot\,]^{\cdot}$

(g) $[\,\bullet \geq \bullet\,]^{\cdot}$

(h) $[\,\mathsf{T} \geq \mathsf{T}\,]^{\cdot}$

(i) $[\,\mathsf{T} \geq \mathsf{F}\,]^{\cdot}$

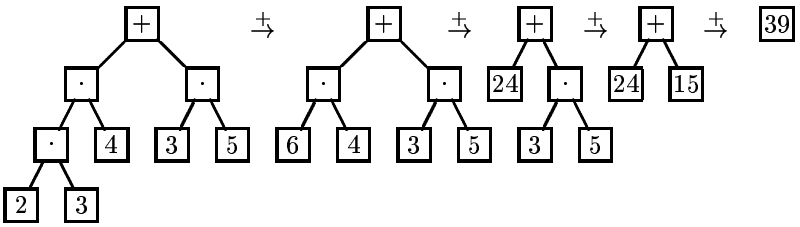
(j) $[\,\mathsf{F} \geq \mathsf{T}\,]^{\cdot}$

(k) $[\,\mathsf{F} \geq \mathsf{F}\,]^{\cdot}$

(l) $[\,\bot \geq \bot\,]^{\cdot}$

2.17 Answers

Answer 2.1.1

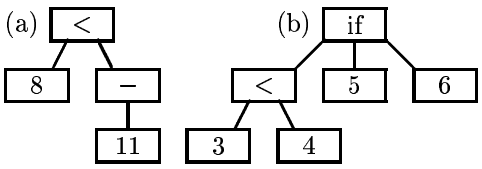


Answer 2.2.1 $\left[\, f_1(\mathsf{x}) \stackrel{+}{\rightarrow} \mathsf{x}^2 - 1 \,\right]^{\cdot}$ and $\left[\, f_5(\mathsf{x},\mathsf{y}) \stackrel{+}{\rightarrow} \mathsf{x}^2 - \mathsf{y}^2 \,\right]^{\cdot}$. Map applies reduction rules (i.e. graph reduction rules) when it computes values on basis of syntax trees. A construct like $[\, f_4(\mathsf{x}) \,]^{\cdot}$ cannot occur in a syntax tree because Map replaces it by $[\, 2 \cdot \mathsf{x} + 4 \,]^{\cdot}$ when it constructs the syntax tree. Therefore, Map needs no reduction rule for $[\, f_4(\mathsf{x}) \,]^{\cdot}$.

Another way to say this is: $[\, f_4(\mathsf{x}) \doteq 2 \cdot \mathsf{x} + 4 \,]$ corresponds to a macro reduction rule, not to a graph reduction rule, and "reduction rule" means "graph reduction rule" in this book.

Answer 2.3.1 $[\,\mathsf{T}\,]^{\cdot}$, $[\,\mathsf{F}\,]^{\cdot}$, $[\,\mathsf{F}\,]^{\cdot}$, and $[\,\mathsf{F}\,]^{\cdot}$.

Answer 2.5.1 (a) $[\,\mathsf{F}\,]^{\cdot}$ and (b) $[\,5\,]^{\cdot}$.



Answer 2.6.1 (a) $[\,0\,]^{\cdot}$, (b) $[\,2\,]^{\cdot}$, (c) $[\,4\,]^{\cdot}$, (d) $[\,6\,]^{\cdot}$, (e) $[\,2 \cdot \mathsf{n}\,]^{\cdot}$, (f) computation never ends.

Answer 2.9.1 (a) $[\,\mathsf{T}\,]^{\cdot}$, (b) $[\,\mathsf{F}\,]^{\cdot}$, (c) $[\,\bot\,]^{\cdot}$, (d) $[\,\mathsf{T}\,]^{\cdot}$, and (e) $[\,\bot\,]^{\cdot}$.

Answer 2.12.1 $[\![\, \mathsf{n} \equiv 0 \,]\!]$ is a counterexample. For this value of $[\![\, \mathsf{n} \,]\!]$, the value of $[\![\, \mathsf{n!} \,]\!]$ is $[\![\, 1 \,]\!]$ whereas the value of $[\![\, \mathsf{n} \cdot (\mathsf{n} - 1)! \,]\!]$ is $[\![\, \bot \,]\!]$.

Answer 2.13.1 (a) $[\![\, 2 \,]\!]$, (b) $[\![\, 1 \,]\!]$, (c) $[\![\, -3 \,]\!]$, and (d) $[\![\, 2 \,]\!]$.

Answer 2.14.1 (a) $[\![\, \bullet \,]\!]$, and (b) $[\![\, \bot \,]\!]$.

Answer 2.15.1 (a) $[\![\, 4 \,]\!]$ and (b) $[\![\, \bot \,]\!]$.

Answer 2.16.1 What are the values of the following:

(a) $[\![\, \mathsf{T} \,]\!]$

(b) $[\![\, \mathsf{T} \,]\!]$

(c) $[\![\, \mathsf{F} \,]\!]$

(d) $[\![\, \bullet \,]\!]$

(e) $[\![\, \bullet \,]\!]$

(f) $[\![\, \bot \,]\!]$

(g) $[\![\, \bullet \,]\!]$

(h) $[\![\, \mathsf{T} \,]\!]$

(i) $[\![\, \mathsf{T} \,]\!]$

(j) $[\![\, \mathsf{F} \,]\!]$

(k) $[\![\, \mathsf{T} \,]\!]$

(l) $[\![\, \bot \,]\!]$

Chapter 3

Decimal fractions

3.1 Overview

I assume you know what a real number is. Just to recall, the ratio $[\boxed{\pi}]^{\cdot}$ between the perimeter and diameter of a circle in Euclidean space is a real number. The value of $[\pi]^{\cdot}$ is close to $[3.14159]^{\cdot}$.

The integer $[117]^{\cdot}$ is another example of a real number and so is the decimal fraction $[2.345]^{\cdot}$. Here you have an overview:

Number	Real number	Decimal fraction	Integer
$[\pi]^{\cdot}$	yes	no	no
$[2.345]^{\cdot}$	yes	yes	no
$[117.000]^{\cdot}$	yes	yes	yes

In general, I call a number a *decimal fraction* if you can express it on the form

$$[\mathsf{m} \cdot 10^{\mathsf{e}}]^{\cdot}$$

where $[\mathsf{m}]^{\cdot}$ and $[\mathsf{e}]^{\cdot}$ are integers. As an example, you have

$$[2.345 \equiv 2345 \cdot 10^{-3}]$$

so $[2.345]^{\cdot}$ is a decimal fraction. There are no integers $[\mathsf{m}]^{\cdot}$ and $[\mathsf{e}]^{\cdot}$ for which

$$[\pi \equiv \mathsf{m} \cdot 10^{\mathsf{e}}]^{\cdot}$$

so $[\pi]^{\cdot}$ is no decimal fraction. I devote this chapter to decimal fractions.

$[\pi]^{\cdot}$ pi
decimal fraction

3.2 Infinity

In addition to the decimal fractions you can write on form

$$[\,\mathsf{m}\cdot 10^{\mathsf{e}}\,]^{\cdot}$$

I introduce two more:

$$\left[\,\boxed{+\infty}\,\right]^{\cdot}$$

and

$$\left[\,\boxed{-\infty}\,\right]^{\cdot}.$$

These two numbers are particularly useful in connection with comparisons. Here you have some examples:

$$\begin{array}{llrll}
[\,2.345 & < & +\infty & \equiv & \mathsf{T}\,] \\
[\,2.345 & \leq & +\infty & \equiv & \mathsf{T}\,] \\
[\,+\infty & < & +\infty & \equiv & \mathsf{F}\,] \\
[\,+\infty & \leq & +\infty & \equiv & \mathsf{T}\,] \\
[\,-\infty & < & 2.345 & \equiv & \mathsf{T}\,] \\
[\,-\infty & < & +\infty & \equiv & \mathsf{T}\,]
\end{array}$$

Unary plus applied to an infinity yields the infinity itself and unary minus changes the sign of the infinity:

$$\begin{array}{lcl}
[\,+(+\infty) & \equiv & +\infty\,] \\
[\,+(-\infty) & \equiv & -\infty\,] \\
[\,-(+\infty) & \equiv & -\infty\,] \\
[\,-(-\infty) & \equiv & +\infty\,]
\end{array}$$

Actually, $[\,\boxed{\infty}\,]^{\cdot}$ stands for plus infinity, and $[\,+\infty\,]^{\cdot}$ and $[\,-\infty\,]^{\cdot}$ are unary plus and unary minus, respectively, applied to $[\,\infty\,]^{\cdot}$.

I choose to make all other arithmetic operations than unary plus and minus return $[\,\bullet\,]^{\cdot}$ when applied to $[\,+\infty\,]^{\cdot}$ and $[\,-\infty\,]^{\cdot}$:

$$\begin{array}{lcrcl}
[\,+\infty & + & 4 & \equiv & \bullet\,] \\
[\,(-\infty) & \cdot & -\infty & \equiv & \bullet\,]
\end{array}$$

Exercise 3.2.1 I define

$$\left[\,\boxed{\min(\mathsf{x},\mathsf{y})} \doteq \mathsf{x} < \mathsf{y}\left\{\begin{array}{l}\mathsf{x}\\ \mathsf{y}\end{array}\right.\right].$$

Compute the following

$[\,+\infty\,]^{\cdot}$	plus infinity
$[\,-\infty\,]^{\cdot}$	minus infinity
$[\,\infty\,]^{\cdot}$	infinity
$[\,\min(\mathsf{x},\mathsf{y})\,]^{\cdot}$	minimum x comma y end

(a) $[\,\min(2,5)\,]^{\cdot}$.

(b) $[\,\min(5,2)\,]^{\cdot}$.

(c) $[\,\min(2,2)\,]^{\cdot}$.

(d) $[\,\min(2,+\infty)\,]^{\cdot}$.

(e) $[\,\min(2,-\infty)\,]^{\cdot}$.

3.3 Exact and floating fractions

If you ask a mathematician to compute

$$[\,87654321 + 1111.1111\,]^{\cdot}$$

then you probably get the answer

$$[\,87655432.1111\,]^{\cdot}.$$

However, if you do the computation on an 8-digit pocket calculator, you are likely to get

$$[\,87655432\,]^{\cdot}$$

because the calculator can give you no more than 8 digits of result. The first result is the correct sum of two exact decimal fractions whereas the second result is the correct sum of two 8-digit numbers. I will write the two computations as

$[\,87654321 + 1111.1111 \equiv 87655432.1111\,]^{\cdot}$, and

$[\,87654321\mathrm{F} + 1111.1111\mathrm{F} \equiv 87655432\mathrm{F}\,]^{\cdot}$, respectively.

In general, I will distinguish between two kinds of decimal fractions, namely *exact fractions* like $[\,1111.1111\,]^{\cdot}$ and *floating fractions* like $[\,1111.1111\mathrm{F}\,]^{\cdot}$. The "F" in $[\,1111.1111\mathrm{F}\,]^{\cdot}$ refers to "float". Exact fractions have nice mathematical properties and floating fractions have nice computational properties.

A floating fraction is a structure that has a *magnitude* and a *precision*. As an example, the magnitude of $[\,1111.1111\mathrm{F}\,]^{\cdot}$ is $[\,1111.1111\,]^{\cdot}$ and the precision is $[\,8\,]^{\cdot}$. The precision is a count of the digits in the numeral. Zeros with a stroke

exact fraction
floating fraction
magnitude
precision

like $[\,\emptyset\,]^\circ$ should not be counted when finding the precision. Here you have some examples of magnitudes and precisions:

Floating fraction	Magnitude	Precision
2.345F	2.345	4
2.3450F	2.345	5
0.0012F	0.0012	5
$\emptyset.\emptyset\emptyset$12F	0.0012	2
12000F	12000	5
120$\emptyset\emptyset$F	12000	3
0.00F	0	3

Here you have an example of an exact fraction:

$$[\,2.345\,]^\cdot.$$

You can recognise exact fractions in that they contain no $[\,\mathtt{F}\,]^\circ$. The exact fraction $[\,2.345\,]^\cdot$ has magnitude $[\,2.345\,]^\cdot$ and precision $[\,+\infty\,]^\cdot$. In general, exact fractions have themselves as magnitude and have precision $[\,+\infty\,]^\cdot$.

The magnitude of a floating fraction is always an exact fraction and the precision of a floating fraction is always a positive integer. I have deliberately chosen to forbid floating fractions to have zero precision, negative precision, and non-integral precision.

$[\,+\infty\,]^\cdot$ and $[\,-\infty\,]^\cdot$ are exact fractions. Later, I introduce floating fractions that have magnitude $[\,+\infty\,]^\cdot$ and $[\,-\infty\,]^\cdot$ and finite precision.

I refer to exact fractions that are also whole numbers as *integers*, and to integers that are non-negative as *natural numbers*. As an example $[\,2.0\,]^\cdot$ is an integer and $[\,2.1\,]^\cdot$ and $[\,2.0\mathtt{F}\,]^\cdot$ are not in my terminology. If you want to regard $[\,2.0\mathtt{F}\,]^\cdot$ as an integer, you may use your notational freedom to do so.

Exercise 3.3.1 What are the magnitudes and precisions of the following numbers. Which of the numbers are integers?

(a)	117F
(b)	117
(c)	117.0
(d)	$\emptyset.\emptyset$100F
(e)	−2.34F
(f)	$+\infty$
(g)	$-\infty$

integer
natural number

3.4 Predicates

As I mentioned in Section 2.5, the predicates $\left[\boxed{x = y}\right]^{\cdot}$, $\left[\boxed{x \neq y}\right]^{\cdot}$, $\left[\boxed{x < y}\right]^{\cdot}$, $\left[\boxed{x \leq y}\right]^{\cdot}$, $\left[\boxed{x > y}\right]^{\cdot}$, and $\left[\boxed{x \geq y}\right]^{\cdot}$ compare magnitudes. When you read that in Section 2.5 you may not have suspected the word "magnitude" to have a special meaning, but now you should know what I mean: the predicates disregard precision. As an example,

$$[\ 2.0\mathrm{F} < 3.00\mathrm{F}\ ]$$

is true because the magnitude of $[\ 2.0\mathrm{F}\ ]^{\cdot}$ (which is $[\ 2\ ]^{\cdot}$) is strongly less than the magnitude of $[\ 3.00\mathrm{F}\ ]^{\cdot}$ (which is $[\ 3\ ]^{\cdot}$). Here you have some examples:

x = y	8	8.0F	7	7F
8	T	T	F	F
8.0F	T	T	F	F
7	F	F	T	T
7F	F	F	T	T

x < y	8	8.0F	7	7F
8	F	F	F	F
8.0F	F	F	F	F
7	T	T	F	F
7F	T	T	F	F

x ≤ y	8	8.0F	7	7F
8	T	T	F	F
8.0F	T	T	F	F
7	T	T	T	T
7F	T	T	T	T

This gives you yet another difference between $[\ x \equiv y\ ]^{\cdot}$ and $[\ x = y\ ]^{\cdot}$: $[\ 8 = 8.0\mathrm{F}\ ]^{\cdot}$ is true, $[\ 8 \equiv 8.0\mathrm{F}\ ]^{\cdot}$ fails.

3.5 Rounding

In ancient Greek mythology, Procrustes was a bad guy with two beds and a bad habit. One bed was long and one was short. When Procrustes got a tall visitor,

$[\ x = y\ ]^{\cdot}$ x computational equal y
$[\ x \neq y\ ]^{\cdot}$ x computational unequal y
$[\ x < y\ ]^{\cdot}$ x strongly less than y
$[\ x \leq y\ ]^{\cdot}$ x weakly less than y
$[\ x > y\ ]^{\cdot}$ x strongly greater than y
$[\ x \geq y\ ]^{\cdot}$ x weakly greater than y

he choped off head and feet to make him fit the short bed and when he got a short visitor, he stretched him to fit the long bed. I now introduce an operation

$$\left[\boxed{\mathsf{x@y}}\right]$$

that does to numbers what Procrustes did to visitors. More precisely, $[\,\mathsf{x@y}\,]$ converts $[\,\mathsf{x}\,]$ into a number that has precision $[\,\mathsf{y}\,]$.

I first give an example of stretching a number. $[\,1.2\mathrm{F}\,]$ has magnitude $[\,1.2\,]$ and precision $[\,2\,]$. $[\,1.2\mathrm{F@}5\,]$ has the same magnitude but precision $[\,5\,]$, so you have:

$$[\,1.2\mathrm{F@}5 \equiv 1.2000\mathrm{F}\,].$$

As another example, $[\,1.2\mathrm{F@}\infty\,]$ has magnitude $[\,1.2\,]$ and precision $[\,\infty\,]$, so

$$[\,1.2\mathrm{F@}\infty \equiv 1.2\,].$$

Here you have some examples of cutting numbers:

$$[\,1.234\mathrm{F@}3 \equiv 1.23\mathrm{F}\,], \text{ and}$$

$$[\,1.236\mathrm{F@}3 \equiv 1.24\mathrm{F}\,].$$

In general, when cutting a number to lower precision, the magnitude is rounded to the nearest values of that precision.

This gives a problem when a value is equally distant from the two nearest values of the given precision. As an example,

$$[\,1.235\mathrm{F}\,]$$

is equally distant from

$$[\,1.23\mathrm{F}\,]$$

and

$$[\,1.24\mathrm{F}\,].$$

In this situation I use the *round to even* convention to break the tie. The round to even convention says that in case of a tie, one shall prefer the candidate whose last digit is even. In

$$[\,1.23\mathrm{F}\,]$$

the last digit is $[\,3\,]$ which is odd and in

$$[\,1.24\mathrm{F}\,]$$

$[\,\mathsf{x@y}\,]$ x round y
round to even

the last digit is $[\,4\,]^{\cdot}$ which is even. Hence,

$$[\,1.235\mathrm{F}@3 \equiv 1.24\mathrm{F}\,]^{\cdot}$$

so round to even results in a round up in this case. As an example of the opposite, i.e. a situation where round to even results in a round down, you have

$$[\,1.225\mathrm{F}@3 \equiv 1.22\mathrm{F}\,]^{\cdot}.$$

As you can see, the round to even convention rounds half of the ties up and half of the ties down, which is convenient in approximate computations.

In addition to round to even, I also introduce round to $[\,+\infty\,]^{\cdot}$ $\left[\,\boxed{\mathsf{x}@_{+}\mathsf{y}}\,\right]^{\cdot}$, round to $[\,-\infty\,]^{\cdot}$ $\left[\,\boxed{\mathsf{x}@_{-}\mathsf{y}}\,\right]^{\cdot}$ and round to $[\,0\,]^{\cdot}$ $\left[\,\boxed{\mathsf{x}@_{0}\mathsf{y}}\,\right]^{\cdot}$. I think you can understand what they do from these examples:

x	x@3	$\mathsf{x}@_{+}3$	$\mathsf{x}@_{-}3$	$\mathsf{x}@_{0}3$
1.234F	1.23F	1.24F	1.23F	1.23F
1.236F	1.24F	1.24F	1.23F	1.23F
−1.234F	−1.23F	−1.23F	−1.24F	−1.23F
−1.236F	−1.24F	−1.23F	−1.24F	−1.23F

I follow the so-called *IEEE floating point standard* in considering these four kinds of rounding, but I depart from that standard on other points. As an important example I consider decimal fractions (base $[\,10\,]^{\cdot}$) whereas the IEEE standard considers so-called "binary fractions" (base $[\,2\,]^{\cdot}$).

Sometimes I will use $[\,\mathsf{x}@_{2}\mathsf{y}\,]^{\cdot}$ as a synonym for $[\,\mathsf{x}@\mathsf{y}\,]^{\cdot}$ to emphasise the similarity to $[\,\mathsf{x}@_{+}\mathsf{y}\,]^{\cdot}$, $[\,\mathsf{x}@_{-}\mathsf{y}\,]^{\cdot}$ and $[\,\mathsf{x}@_{0}\mathsf{y}\,]^{\cdot}$:

$$\left[\,\boxed{\mathsf{x}@_{2}\mathsf{y}} \doteq \mathsf{x}@\mathsf{y}\,\right].$$

Rounding may be applied to infinities. As an example, $[\,(-\infty)@3\,]^{\cdot}$ is a floating fraction with magnitude $[\,-\infty\,]^{\cdot}$ and precision $[\,3\,]^{\cdot}$.

$[\,\mathsf{x}@_{+}\mathsf{y}\,]^{\cdot}$	x round up y
$[\,\mathsf{x}@_{-}\mathsf{y}\,]^{\cdot}$	x round down y
$[\,\mathsf{x}@_{0}\mathsf{y}\,]^{\cdot}$	x round zero y
	IEEE floating point standard
$[\,\mathsf{x}@_{2}\mathsf{y}\,]^{\cdot}$	x round even y

Exercise 3.5.1 Compute the following:

(a)	1.4F@1
(b)	1.5F@1
(c)	1.6F@1
(d)	2.4F@1
(e)	2.5F@1
(f)	2.6F@1
(g)	$(-2.3)@_2 1$
(h)	$(-2.3)@_+ 1$
(i)	$(-2.3)@_- 1$
(j)	$(-2.3)@_0 1$
(k)	0.995F@2
(l)	1.23F@3

3.6 Precision

I use $\left[\boxed{\#\mathsf{x}}\right]^{\cdot}$ to denote the precision of $[\,\mathsf{x}\,]^{\cdot}$. As an example,

$[\,\#1.23\mathrm{F} \equiv 3\,]$, and

$[\,\#1.23 \equiv +\infty\,]$.

Exercise 3.6.1 Compute the following:

(a)	#0.0100F
(b)	#10000F
(c)	#(1.288523@3)
(d)	$\#+\infty$
(e)	$\#-\infty$
(f)	$\#(+\infty@3)$

3.7 Exact arithmetic operations

I will refer to $\left[\boxed{\mathsf{x}+\mathsf{y}}\right]^{\cdot}$, $\left[\boxed{\mathsf{x}-\mathsf{y}}\right]^{\cdot}$, and $\left[\boxed{\mathsf{x}\cdot\mathsf{y}}\right]^{\cdot}$ as *exact arithmetic operations*. If you apply one of these operations to exact fractions, the result will be an exact fraction. Here you have some examples:

$[\,10^5 + 10^{-5} \equiv 100000.00001\,]$,

$[\,2.3 - 3.4 \equiv -1.1\,]$, and

$[\,\#\mathsf{x}\,]^{\cdot}$	precision x
$[\,\mathsf{x}+\mathsf{y}\,]^{\cdot}$	x plus y
$[\,\mathsf{x}-\mathsf{y}\,]^{\cdot}$	x minus y
$[\,\mathsf{x}\cdot\mathsf{y}\,]^{\cdot}$	x times y
	exact arithmetic operation

$$[\, 1.1111 \cdot 1.111 \equiv 1.2344321 \,].$$

I make the convention (following the IEEE standard) that if the operations are applied to numbers of the same precision, then the operation is performed with precision $[\, \infty \,]$ followed by rounding. Here you have an example:

$$\begin{aligned} [\, 9.81\mathrm{F} + \emptyset.311\mathrm{F} &= (9.81\mathrm{F}@\infty + \emptyset.311\mathrm{F}@\infty)@\#9.81\mathrm{F} \\ &= (9.81 + \emptyset.311)@3 \\ &= 10.121@3 \\ &= 10.1\mathrm{F}\,] \end{aligned}$$

As you can see, $[\, x + y \,]$ uses round to even. Here you have addition with round to $[\, +\infty \,]$:

$$\begin{aligned} [\, 9.81\mathrm{F} +_+ \emptyset.311\mathrm{F} &= (9.81\mathrm{F}@\infty + \emptyset.311\mathrm{F}@\infty)@_+\#9.81\mathrm{F} \\ &= (9.81 + \emptyset.311)@_+3 \\ &= 10.121@_+3 \\ &= 10.2\mathrm{F}\,] \end{aligned}$$

In general, you have the following $[\, 15 \,]$ operations:

$$\begin{array}{ccccc} [\, x + y \,] & [\, x +_2 y \,] & [\, x +_+ y \,] & [\, x +_- y \,] & [\, x +_0 y \,] \\ [\, x - y \,] & [\, x -_2 y \,] & [\, x -_+ y \,] & [\, x -_- y \,] & [\, x -_0 y \,] \\ [\, x \cdot y \,] & [\, x \cdot_2 y \,] & [\, x \cdot_+ y \,] & [\, x \cdot_- y \,] & [\, x \cdot_0 y \,] \end{array}$$

$[\, x +_2 y \,]$, $[\, x -_2 y \,]$, and $[\, x \cdot_2 y \,]$ are synonyms for $[\, x + y \,]$, $[\, x - y \,]$, and $[\, x \cdot y \,]$, respectively.

Rounding has no effect on exact decimal fractions:

$$[\, 10^5 +_+ 10^{-5} \equiv 100000.00001 \,].$$

I make the convention that operations that mix a floating and an exact fraction are done with precision $[\, \infty \,]$ followed by rounding to the precision of the floating fraction. Here you have an example:

$$\begin{aligned} 1.23\mathrm{F} +_+ 10^{-10} &\equiv (1.23\mathrm{F}@\infty + 10^{-10})@_+\#1.23\mathrm{F} \\ &\equiv (1.23 + 10^{-10})@_+3 \\ &\equiv 1.2300000001@_+3 \\ &\equiv 1.24\mathrm{F} \end{aligned}$$

I make the convention that operations that mix floating fractions of different precision yield $[\, \bullet \,]$:

$$[\, 1.0\mathrm{F} + 2.00\mathrm{F} \equiv \bullet \,].$$

Here you have some further examples:

$x + y$	7	8	7.0F	8.00F
7	14	15	14F	15.0F
8	15	16	15F	16.0F
7.0F	14F	15F	14F	$\bullet$
8.00F	15.0F	16.0F	$\bullet$	16.0F

x − y	7	8	7.0F	8.00F
7	0	−1	0.0F	−1.00F
8	1	0	1.0F	0.00F
7.0F	0.0F	−1.0F	0.0F	•
8.00F	1.00F	0.00F	•	0.00F

x · y	7	8	7.0F	8.00F
7	49	56	49F	56.0F
8	56	64	56F	64.0F
7.0F	49F	56F	49F	•
8.00F	56.0F	64.0F	•	64.0F

Exercise 3.7.1 Compute the following:

(a)	2.00F − 2
(b)	2.00F − 2.000F
(c)	1.0F − 0.005
(d)	∅.50F + ∞
(e)	∅.∅50F − 2.0F
(f)	∅.∅50F $-_+$ 2.0F
(g)	∅.∅50F $-_-$ 2.0F
(h)	∅.∅50F $-_0$ 2.0F

3.8 Division

Map pretends that it computes $\left[\boxed{\text{1.00F/ 3.00F}}\right]^{\cdot}$ thus:

$$\begin{aligned} [\,1.00F/3.00F &\equiv (0.3333\cdots)@\#1.00F \\ &\equiv (0.3333\cdots)@3 \\ &\equiv 0.333F\,]^{\circ} \end{aligned}$$

In other words, Map pretends that it computes the quotient with infinite precision and then rounds correctly. In reality, Map cannot compute infinitely many digits, so it has to do something else behind the scenes. You, however, do not need to bother how Map does a division as long as you can always predict the result.

If $[\,x\,]^{\cdot}$ and $[\,y\,]^{\cdot}$ have different precision, if one or both has precision $[\,\infty\,]^{\cdot}$, or if $[\,y = 0\,]^{\cdot}$ then $[\,x/y \equiv \bullet\,]^{\cdot}$. Here you have some examples:

$$\begin{aligned} [\,10/5 &\equiv \bullet\,] \\ [\,10.0F/5.0F &\equiv \bullet\,] \\ [\,10.0F/5 &\equiv \bullet\,] \\ [\,10.0F/0.00F &\equiv \bullet\,] \end{aligned}$$

Like the other arithmetic operators, $[\,x/y\,]^{\cdot}$ has four rounding modes, $[\,x/_2y\,]^{\cdot}$, $[\,x/_+y\,]^{\cdot}$, $[\,x/_-y\,]^{\cdot}$, and $[\,x/_0y\,]^{\cdot}$, where $[\,x/_2y\,]^{\cdot}$ is the same operation as $[\,x/y\,]^{\cdot}$.

$[\,x/y\,]^{\cdot}$ x divide y

Exercise 3.8.1 Compute the following:

(a) $[\![\, 2.27\mathrm{F}/_0 2.00\mathrm{F} \,]\!]$.

(b) $[\![\, 2.23\mathrm{F}/2 \,]\!]$.

3.9 Integer division

I presented *integer division* in Section 2.13. I now generalise integer division to decimal fractions.

If $[\![\, x \,]\!]$ is a decimal fraction and $[\![\, y \,]\!]$ is a positive decimal fraction, then there is one and only one integer $[\![\, q \,]\!]$ and one and only one exact decimal fraction $[\![\, r \,]\!]$ such that

- $[\![\, x = q \cdot y + r \,]\!]$, and
- $[\![\, 0 \leq r < y \,]\!]$.

I denote these values $\left[\!\left[\, \boxed{x \mathbin{//} y} \,\right]\!\right]$ and $\left[\!\left[\, \boxed{x \mathbin{\%} y} \,\right]\!\right]$, respectively. These operations disregard precision. Here you have some examples:

$$\begin{array}{rcr@{\quad}rcr}
[\![\, 7 \mathbin{//} 3 & \equiv & 2 \,]\!] & [\![\, 7 \mathbin{\%} 3 & \equiv & 1 \,]\!] \\
[\![\, 7.0\mathrm{F} \mathbin{//} 3.00\mathrm{F} & \equiv & 2 \,]\!] & [\![\, 7.0\mathrm{F} \mathbin{\%} 3.00\mathrm{F} & \equiv & 1 \,]\!] \\
[\![\, (-7.23) \mathbin{//} 3 & \equiv & -3 \,]\!] & [\![\, (-7.23) \mathbin{\%} 3 & \equiv & 1.77 \,]\!] \\
[\![\, 7 \mathbin{//} (-1) & \equiv & \bullet \,]\!] & [\![\, 7 \mathbin{\%} (-1) & \equiv & \bullet \,]\!] \\
[\![\, 1 \mathbin{//} 0.98 & \equiv & 1 \,]\!] & [\![\, 1 \mathbin{\%} 0.98 & \equiv & 0.02 \,]\!]
\end{array}$$

You may interpret $[\![\, x \mathbin{//} y \,]\!]$ as integer division with round to $[\![\, -\infty \,]\!]$. As an example, you may interpret $[\![\, 7 \mathbin{//} 3 \,]\!]$ as $[\![\, 2.3333\cdots \,]\!]^\circ$ rounded towards $[\![\, -\infty \,]\!]$. In analogy with the other arithmetic operations, I define four versions of integer division:

$$\begin{array}{cccc}
[\![\, x \mathbin{//_2} y \,]\!] & [\![\, x \mathbin{//_+} y \,]\!] & [\![\, x \mathbin{//_-} y \,]\!] & [\![\, x \mathbin{//_0} y \,]\!] \\
[\![\, x \mathbin{\%_2} y \,]\!] & [\![\, x \mathbin{\%_+} y \,]\!] & [\![\, x \mathbin{\%_-} y \,]\!] & [\![\, x \mathbin{\%_0} y \,]\!]
\end{array}$$

$[\![\, x \mathbin{//_-} y \,]\!]$ and $[\![\, x \mathbin{\%_-} y \,]\!]$ are synonyms for $[\![\, x \mathbin{//} y \,]\!]$ and $[\![\, x \mathbin{\%} y \,]\!]$, respectively. For all decimal fractions $[\![\, x \,]\!]$ and all positive decimal fractions $[\![\, y \,]\!]$ you have:

- $[\![\, x \equiv (x \mathbin{//} y) \cdot y + (x \mathbin{\%} y) \,]\!]$.
- $[\![\, x \equiv (x \mathbin{//_2} y) \cdot y + (x \mathbin{\%_2} y) \,]\!]$.
- $[\![\, x \equiv (x \mathbin{//_+} y) \cdot y + (x \mathbin{\%_+} y) \,]\!]$.
- $[\![\, x \equiv (x \mathbin{//_-} y) \cdot y + (x \mathbin{\%_-} y) \,]\!]$.
- $[\![\, x \equiv (x \mathbin{//_0} y) \cdot y + (x \mathbin{\%_0} y) \,]\!]$.

	integer division
$[\![\, x \mathbin{//} y \,]\!]$	x integer divide y
$[\![\, x \mathbin{\%} y \,]\!]$	x modulo y

Here you have some examples.

$$
\begin{array}{llrllr}
[\,7\,//_2\,3 & \equiv & 2\,] & [\,7\,\%_2\,3 & \equiv & 1\,] \\
[\,7\,//_+\,3 & \equiv & 3\,] & [\,7\,\%_+\,3 & \equiv & -2\,] \\
[\,7\,//_-\,3 & \equiv & 2\,] & [\,7\,\%_-\,3 & \equiv & 1\,] \\
[\,7\,//_0\,3 & \equiv & 2\,] & [\,7\,\%_0\,3 & \equiv & 1\,] \\
[\,(-7)\,//_2\,3 & \equiv & -2\,] & [\,(-7)\,\%_2\,3 & \equiv & -1\,] \\
[\,(-7)\,//_+\,3 & \equiv & -2\,] & [\,(-7)\,\%_+\,3 & \equiv & -1\,] \\
[\,(-7)\,//_-\,3 & \equiv & -3\,] & [\,(-7)\,\%_-\,3 & \equiv & 2\,] \\
[\,(-7)\,//_0\,3 & \equiv & -2\,] & [\,(-7)\,\%_0\,3 & \equiv & -1\,]
\end{array}
$$

I choose to let all the constructs above equal $[\,\bullet\,]$ when the second argument is zero or negative, so you have e.g. $[\,2\,//\,0 \equiv \bullet\,]$ and $[\,2\,\%\,(-1) \equiv \bullet\,]$.

Exercise 3.9.1 Compute the following:

(a)	$117.123\,//\,1$
(b)	$117.123\,\%\,1$
(c)	$-117.123\,//\,1$
(d)	$-117.123\,\%\,1$
(e)	$117.123\,//_+\,1$
(f)	$117.123\,\%_+\,1$
(g)	$117.5\,//_2\,1$
(h)	$117.5\,\%_2\,1$
(i)	$112.5\,//_2\,1$
(j)	$112.5\,\%_2\,1$

3.10 Power

I choose to define $\left[\,\boxed{x^y}\,\right]$ such that the construct raises $[\,x\,]$ to power $[\,y\,]$ in the following cases:

(a) $[\,x\,]$ and $[\,y\,]$ are floating fractions with equal precision and $[\,x\,]$ is positive. In this case $[\,x^y\,]$ has the same precision as $[\,x\,]$ and $[\,y\,]$.

(b) $[\,x\,]$ is a floating fraction and $[\,y\,]$ is an exact fraction or vice versa, and $[\,x\,]$ is positive. In this case $[\,x^y\,]$ has the same precision as the floating fraction.

(c) $[\,x\,]$ is a decimal fraction and $[\,y\,]$ is a natural number. In this case $[\,x^y\,]$ has the same precision as $[\,x\,]$. In particular, $[\,x^0 \equiv 1@\#x\,]$.

(d) $[\,x = 10\,]$ and $[\,y\,]$ is an integer. In this case $[\,x^y\,]$ has precision $[\,\infty\,]$.

$[\,x^y\,]$ x power y end

Above, (a) and (b) are approximate computations. (c) is $[\,x\,]^{\cdot}$ multiplied $[\,y\,]^{\cdot}$ times by itself. (d) allows to write numbers like

$$[\,1.23\mathrm{F}\cdot 10^{-5}\,]^{\cdot}.$$

Here you have some examples:

$$\begin{aligned}
[\,2.0\mathrm{F}^{4.0\mathrm{F}} &\equiv 16\mathrm{F}\,]\\
[\,2.0\mathrm{F}^{8.0\mathrm{F}} &\equiv 26\emptyset\mathrm{F}\,]\\
[\,2^{-3} &\equiv \bullet\,]\\
[\,10^{-3} &\equiv 0.001\,]\\
[\,2.0\mathrm{F}^{8.00\mathrm{F}} &\equiv \bullet\,]
\end{aligned}$$

3.11 Mantissa and exponent

You can write any positive decimal fraction $[\,x\,]^{\cdot}$ on the form

$$[\,m\cdot 10^{e}\,]^{\cdot}$$

where

$$[\,1\le m<10\,]^{\cdot}$$

and where $[\,e\,]^{\cdot}$ is an integer, and you can only do it one way for each $[\,x\,]^{\cdot}$. I will refer to $[\,m\,]^{\cdot}$ and $[\,e\,]^{\cdot}$ as the *mantissa* and *exponent* of $[\,x\,]^{\cdot}$, respectively. I define $[\,0\,]^{\cdot}$ to have mantissa $[\,0\,]^{\cdot}$ and exponent $[\,0\,]^{\cdot}$, $[\,+\infty\,]^{\cdot}$ to have mantissa $[\,+\infty\,]^{\cdot}$ and exponent $[\,0\,]^{\cdot}$, and $[\,-\infty\,]^{\cdot}$ to have mantissa $[\,-\infty\,]^{\cdot}$ and exponent $[\,0\,]^{\cdot}$. For negative decimal fractions I define the mantissa and exponent to be the numbers $[\,m\,]^{\cdot}$ and $[\,e\,]^{\cdot}$ for which

$$[\,x=m\cdot 10^{e}\,]^{\cdot},\text{ and}$$

$$[\,-10<m\le -1\,]^{\cdot}.$$

The mantissa of a decimal fraction $[\,x\,]^{\cdot}$ has the same precision as $[\,x\,]^{\cdot}$ and the exponent is always an integer, i.e. it has precision $[\,\infty\,]^{\cdot}$. Here you have some examples:

x	mantissa	exponent
117.123	1.17123	2
$\emptyset.\emptyset$12F	1.2F	-2
12000F	1.2000F	4
0	0	0
$+\infty$	$+\infty$	0
$-\infty$	$-\infty$	0
-0.012	-1.2	-2

mantissa
exponent

3.12 Answers

Answer 3.2.1 (a) $[\,2\,]^-$, (b) $[\,2\,]^-$, (c) $[\,2\,]^-$, (d) $[\,2\,]^-$, and (e) $[\,-\infty\,]^-$.

Answer 3.3.1 (b) and (c) are integers, the others are not. The magnitudes and precisions are thus:

(a)	117F	117	3
(b)	117	117	$+\infty$
(c)	117.0	117	$+\infty$
(d)	$\emptyset.\emptyset$100F	0.01	3
(e)	-2.34F	-2.34	3
(f)	$+\infty$	$+\infty$	$+\infty$
(g)	$-\infty$	$-\infty$	$+\infty$

Answer 3.5.1

(a)	1.4F@1	1F
(b)	1.5F@1	2F
(c)	1.6F@1	2F
(d)	2.4F@1	2F
(e)	2.5F@1	2F
(f)	2.6F@1	3F
(g)	$(-2.3)@_2 1$	-2F
(h)	$(-2.3)@_+ 1$	-2F
(i)	$(-2.3)@_- 1$	-3F
(j)	$(-2.3)@_0 1$	-2F
(k)	0.995F@2	1.0F
(l)	1.23F@3	1.23F

In (k) note that there is a tie between the values $[\,0.99\text{F}\,]^-$ and $[\,1.0\text{F}\,]^-$ and that $[\,1.0\text{F}\,]^-$ is the right result since its last digit is even. In (l) note that rounding does not change the number at all, so $[\,\text{x@y}\,]^-$ is not quite as bad as Procrustes.

Answer 3.6.1

(a)	#0.0100F	5
(b)	#10000F	5
(c)	#(1.288523@3)	3
(d)	$\#+\infty$	$+\infty$
(e)	$\#-\infty$	$+\infty$
(f)	$\#(+\infty@3)$	3

Answer 3.7.1

(a)	$2.00\mathtt{F} - 2$	$0.00\mathtt{F}$
(b)	$2.00\mathtt{F} - 2.000\mathtt{F}$	$\bullet$
(c)	$1.0\mathtt{F} - 0.005$	$1.0\mathtt{F}$
(d)	$\emptyset.50\mathtt{F} + \infty$	$\bullet$
(e)	$\emptyset.\emptyset50\mathtt{F} - 2.0\mathtt{F}$	$-2.0\mathtt{F}$
(f)	$\emptyset.\emptyset50\mathtt{F} -_{+} 2.0\mathtt{F}$	$-1.9\mathtt{F}$
(g)	$\emptyset.\emptyset50\mathtt{F} -_{-} 2.0\mathtt{F}$	$-2.0\mathtt{F}$
(h)	$\emptyset.\emptyset50\mathtt{F} -_{0} 2.0\mathtt{F}$	$-1.9\mathtt{F}$

Answer 3.8.1 (a) $[\ 1.13\mathtt{F}\ ]$ and (b) $[\ \bullet\ ]$.

Answer 3.9.1

(a)	$117.123 \mathbin{//} 1$	117
(b)	$117.123 \mathbin{\%} 1$	0.123
(c)	$-117.123 \mathbin{//} 1$	-118
(d)	$-117.123 \mathbin{\%} 1$	0.877
(e)	$117.123 \mathbin{//_{+}} 1$	118
(f)	$117.123 \mathbin{\%_{+}} 1$	-0.877
(g)	$117.5 \mathbin{//_{2}} 1$	118
(h)	$117.5 \mathbin{\%_{2}} 1$	-0.5
(i)	$112.5 \mathbin{//_{2}} 1$	112
(j)	$112.5 \mathbin{\%_{2}} 1$	0.5

Chapter 4

Truth values

4.1 Logical 'and'

[it is sunshine and the birds are singing] is a composite statement composed from the constituent statements [it is sunshine] and [the birds are singing] using the logical connective *and*. I will now analyse 'and' in detail.

If [it is sunshine] and [the birds are singing] are both true, then [it is sunshine and the birds are singing] is true. If [it is sunshine] is false and [the birds are singing] is true, then [it is sunshine and the birds are singing] is false.

You can go on like this and find the truth value of [it is sunshine and the birds are singing] for all four combinations of truth values of [it is sunshine] and [the birds are singing]. Here is a table that shows the truth value of [it is sunshine and the birds are singing] as a function of the truth values of [it is sunshine] and [the birds are singing]. Such a table is known as a *truth table*.

it is sunshine	the birds are singing	it is sunshine and the birds are singing
T	T	T
T	F	F
F	T	F
F	F	F

Here is the table once again, but I have replaced [it is sunshine] by [x] and

and
truth table

[the birds are singing]˙ by [y]˙.

x	y	x and y
T	T	T
T	F	F
F	T	F
F	F	F

I now define $\left[\mathsf{x\ and\ y} \doteq \mathsf{x}\left\{\begin{array}{l}\mathsf{y}\\ \mathsf{F}\end{array}\right.\right]$. You have:

$$\left[\mathsf{T\ and\ T} \equiv \mathsf{T}\left\{\begin{array}{l}\mathsf{T}\\ \mathsf{F}\end{array}\right. \equiv \mathsf{T}\right]^{\cdot},$$

$$\left[\mathsf{T\ and\ F} \equiv \mathsf{T}\left\{\begin{array}{l}\mathsf{F}\\ \mathsf{F}\end{array}\right. \equiv \mathsf{F}\right]^{\cdot},$$

$$\left[\mathsf{F\ and\ T} \equiv \mathsf{F}\left\{\begin{array}{l}\mathsf{T}\\ \mathsf{F}\end{array}\right. \equiv \mathsf{F}\right]^{\cdot}, \text{ and}$$

$$\left[\mathsf{F\ and\ F} \equiv \mathsf{F}\left\{\begin{array}{l}\mathsf{F}\\ \mathsf{F}\end{array}\right. \equiv \mathsf{F}\right]^{\cdot}.$$

Hence, the definition of [x and y]˙ conforms to the truth table.

Since I am in a good mood, I define [it is sunshine ≐ T] and [the birds are singing ≐ T] (actually it is raining cats and dogs outside my window while I write this, but that is irrelevant for a true logician). With these definitions, Map can verify [it is sunshine and the birds are singing].

4.2 $[\mathsf{x} \wedge \mathsf{y}]^{\cdot}$

I will not use [x and y]˙ in the following. Rather, I will use the operator

$$\left[\boxed{\mathsf{x} \wedge \mathsf{y}}\right]^{\cdot}$$

which I describe in the following:

[x ∧ y]˙ is strict so

$[\ \bot \wedge \mathsf{y} \equiv \bot\ ]^{\cdot}$, and

$[\ \mathsf{x} \wedge \bot \equiv \bot\ ]^{\cdot}$.

If [x]˙ and [y]˙ are truth values, then [x ∧ y]˙ has the following values:

$$\frac{[\ \mathsf{T} \wedge \mathsf{T} \equiv \mathsf{T}\ ]^{\cdot},}{[\ \mathsf{x} \wedge \mathsf{y}\ ]^{\cdot}\quad \text{x and y}}$$

$[\,\mathsf{T} \wedge \mathsf{F} \equiv \mathsf{F}\,]$,

$[\,\mathsf{F} \wedge \mathsf{T} \equiv \mathsf{F}\,]$, and

$[\,\mathsf{F} \wedge \mathsf{F} \equiv \mathsf{F}\,]$.

If you look back to Section 2.16 you will see that $[\,\mathsf{x} \wedge \mathsf{y}\,]$ equals $[\,\mathsf{x} \cdot \mathsf{y}\,]$ when $[\,\mathsf{x}\,]$ and $[\,\mathsf{y}\,]$ are truth values. Hardware designers tend to use $[\,\mathsf{x} \cdot \mathsf{y}\,]$ for logical "and".

If $[\,\mathsf{x}\,]$ and $[\,\mathsf{y}\,]$ are numbers of equal precision, then $[\,\mathsf{x} \wedge \mathsf{y}\,]$ denotes the minimum of the two numbers. As an example,

$[\,2 \wedge 5 \equiv 2\,]$.

If you remember that "truth is greater than falsehood" you will see that $[\,\mathsf{x} \wedge \mathsf{y}\,]$ also denotes the minimum of $[\,\mathsf{x}\,]$ and $[\,\mathsf{y}\,]$ when $[\,\mathsf{x}\,]$ and $[\,\mathsf{y}\,]$ are truth values.

If $[\,\mathsf{x}\,]$ is a decimal fraction or a truth value or an exception, then

$[\,\mathsf{x} \wedge \bullet \equiv \bullet\,]$, and

$[\,\bullet \wedge \mathsf{x} \equiv \bullet\,]$.

If $[\,\mathsf{x}\,]$ is a truth value and $[\,\mathsf{y}\,]$ is a decimal fraction then

$[\,\mathsf{x} \wedge \mathsf{y} \equiv \bullet\,]$, and

$[\,\mathsf{y} \wedge \mathsf{x} \equiv \bullet\,]$.

If $[\,\mathsf{x}\,]$ and $[\,\mathsf{y}\,]$ are numbers of different precision, then

$[\,\mathsf{x} \wedge \mathsf{y} \equiv \bullet\,]$.

Exercise 4.2.1 What are the values of the following:

(a) $[\,\mathsf{T} \wedge \mathsf{T} \wedge \mathsf{F}\,]$.

(b) $[\,9 \wedge 3 \wedge 8 \wedge 117\,]$.

(c) $[\,7 \wedge \mathsf{T} \wedge (-1)!\,]$.

4.3 Logical "or"

[either Napoleon wins or Wellington wins] is true if one of the two wins but false if both win and false if neither wins. This is expressed as follows:

Napoleon wins	Wellington wins	either Napoleon wins or Wellington wins
T	T	F
T	F	T
F	T	T
F	F	F

You can express this as $[$ either x or y $\doteq$ if(x, if(y, F, T), if(y, T, F)) $]$. As an example, if [Napoleon wins $\doteq$ F] and [Wellington wins $\doteq$ T] then [either Napoleon wins or Wellington wins] holds. The *either-or* is usually called *exclusive or* in the literature because it excludes the possibility that both constituent statements are true.

$[$ taxes raise and/or prices raise $]^{\cdot}$ is true if taxes raise or prices raise or both, but is false if neither taxes nor prices raise. You can express this as follows:

taxes raise	prices raise	taxes raise and/or prices raise
T	T	T
T	F	T
F	T	T
F	F	F

You can express this as [x and/or y $\doteq$ if(x, T, y)]. As an example, if [taxes raise $\doteq$ T] and [prices raise $\doteq$ T] then [taxes raise and/or prices raise] holds. The *and/or* construct is usually called *inclusive or* in the literature because it includes the possibility that both constituent statements are true.

4.4 $[\mathsf{x} \vee \mathsf{y}]^{\cdot}$

Inclusive "or" happens to be used much more than exclusive "or" in logic and mathematics. Logicians use the symbol

$$\left[\boxed{\mathsf{x} \vee \mathsf{y}}\right]^{\cdot}$$

to denote inclusive or. I describe $[\mathsf{x} \vee \mathsf{y}]^{\cdot}$ in the following.

$[\mathsf{x} \vee \mathsf{y}]^{\cdot}$ is strict so

$[\bot \vee \mathsf{y} \equiv \bot]^{\cdot}$, and

$[\mathsf{x} \vee \bot \equiv \bot]^{\cdot}$.

If $[\mathsf{x}]^{\cdot}$ and $[\mathsf{y}]^{\cdot}$ are truth values, then $[\mathsf{x} \vee \mathsf{y}]^{\cdot}$ has the following values:

$[\mathsf{T} \vee \mathsf{T} \equiv \mathsf{T}]^{\cdot}$,

$[\mathsf{T} \vee \mathsf{F} \equiv \mathsf{T}]^{\cdot}$,

$[\mathsf{F} \vee \mathsf{T} \equiv \mathsf{T}]^{\cdot}$, and

$[\mathsf{F} \vee \mathsf{F} \equiv \mathsf{F}]^{\cdot}$.

	either-or exclusive or and/or inclusive or
$[\mathsf{x} \vee \mathsf{y}]^{\cdot}$	x or y

If you look back to Section 2.16 you will see that $[\,x \vee y\,]$ equals $[\,x + y\,]$ when $[\,x\,]$ and $[\,y\,]$ are truth values. Hardware designers tend to use $[\,x + y\,]$ for logical "or".

If $[\,x\,]$ and $[\,y\,]$ are numbers of equal precision, then $[\,x \vee y\,]$ denotes the maximum of the two numbers. As an example,

$$[\,2 \vee 5 \equiv 5\,].$$

If you remember that "truth is greater than falsehood" you will see that $[\,x \vee y\,]$ also denotes the maximum of $[\,x\,]$ and $[\,y\,]$ when $[\,x\,]$ and $[\,y\,]$ are truth values.

If $[\,x\,]$ is a decimal fraction or a truth value or an exception, then

$$[\,x \vee \bullet \equiv \bullet\,], \text{ and}$$

$$[\,\bullet \vee x \equiv \bullet\,].$$

If $[\,x\,]$ is a truth value and $[\,y\,]$ is a decimal fraction then

$$[\,x \vee y \equiv \bullet\,], \text{ and}$$

$$[\,y \vee x \equiv \bullet\,].$$

If $[\,x\,]$ and $[\,y\,]$ are numbers of different precision, then

$$[\,x \vee y \equiv \bullet\,].$$

Exercise 4.4.1 What are the values of the following:

(a) $[\,3 \vee 5 \vee \infty\,]$.

(b) $[\,3.0\mathrm{F} \vee 5.0\mathrm{F} \vee -\infty\,]$.

4.5 Negation

I use $[\,\boxed{\neg x}\,]$ to denote the negation of $[\,x\,]$, i.e. $[\,\neg x\,]$ is true if $[\,x\,]$ is false and vice versa. If $[\,x\,]$ is a number or an exception then $[\,\neg x \equiv \bullet\,]$. $[\,\neg x\,]$ is strict. In other words,

$$[\,\neg \mathsf{T} \equiv \mathsf{F}\,],$$

$$[\,\neg \mathsf{F} \equiv \mathsf{T}\,],$$

$[\,\neg n \equiv \bullet\,]$ if $[\,n\,]$ is a decimal fraction,

$$[\,\neg \bullet \equiv \bullet\,], \text{ and}$$

$$[\,\neg \bot \equiv \bot\,].$$

Exercise 4.5.1 What are the values of the following:

(a) $[\,\neg\neg\neg\neg \mathsf{T}\,]$.

(b) $[\,\neg\neg\neg\neg\neg \mathsf{T}\,]$.

$[\,\neg x\,]$ not x

4.6 Implication and biimplication

$[\boxed{x \Rightarrow y}]^{\cdot}$, $[\boxed{x \Leftarrow y}]^{\cdot}$, and $[\boxed{x \Leftrightarrow y}]^{\cdot}$ are three more operations on truth values which I mention here for completeness. The truth table reads

x	y	x ⇒ y	x ⇐ y	x ⇔ y
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

If you look back to Section 2.16 you will see that $[\, x \Rightarrow y \,]^{\cdot}$, $[\, x \Leftarrow y \,]^{\cdot}$, and $[\, x \Leftrightarrow y \,]^{\cdot}$ equal $[\, x \leq y \,]^{\cdot}$, $[\, x \geq y \,]^{\cdot}$, and $[\, x = y \,]^{\cdot}$, respectively, when $[\, x \,]^{\cdot}$ and $[\, y \,]^{\cdot}$ are truth values.

Exercise 4.6.1 What are the values of the following:

(a) $[\, 2 < 3 \Rightarrow 3 < 2 \,]^{\cdot}$.

(b) $[\, 3 < 2 \Rightarrow 2 < 3 \,]^{\cdot}$.

4.7 Priority and associativity

I give the logical operators priority between relations and $\left[x \left\{ \begin{array}{c} y \\ z \end{array} \right. \right]^{\cdot}$ as follows:

$$\left[x = y \mathrel{\breve{>}} \neg x \mathrel{\breve{>}} x \wedge y \mathrel{\breve{>}} x \vee y \mathrel{\breve{>}} x \Rightarrow y \mathrel{\breve{=}} x \Leftarrow y \mathrel{\breve{=}} x \Leftrightarrow y \mathrel{\breve{>}} x \left\{ \begin{array}{c} y \\ z \end{array} \right. \right].$$

I make $[\, x \wedge y \,]^{\cdot}$ and $[\, x \vee y \,]^{\cdot}$ left associative like most other operators:

$[\, x \wedge y \wedge z \mathrel{\breve{\rightarrow}} (x \wedge y) \wedge z \,]$, and

$[\, x \vee y \vee z \mathrel{\breve{\rightarrow}} (x \vee y) \vee z \,]$.

Exercise 4.7.1 What are the values of the following:

(a) $[\, 2 = 3 \vee \neg 2 = 3 \,]^{\cdot}$.

(b) $[\, 2 = 3 \Rightarrow 2 = 4 \,]^{\cdot}$.

$[\, x \Rightarrow y \,]^{\cdot}$ x implies y
$[\, x \Leftarrow y \,]^{\cdot}$ x implied by y
$[\, x \Leftrightarrow y \,]^{\cdot}$ x if and only if y

4.8 Associativity of relations

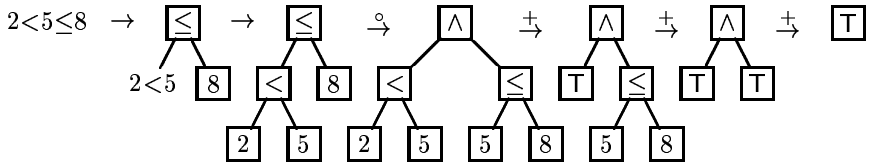
I state the associativity of relations like $[\,<\,]^\circ$ and $[\,\leq\,]^\circ$ by an associativity rule together with a *term reduction rule*:

$$[\,x < y \leq z \mathrel{\breve{\to}} (x < y) \leq z\,], \text{ and}$$

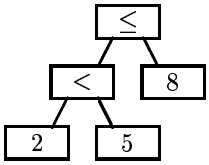
$$\Big[\,x < y \leq z \overset{\circ}{\to} x < y \wedge y \leq z\,\Big].$$

$[\,<\,]^\circ$ and $[\,\leq\,]^\circ$ have the same priority, so the associativity rule $[\,x < y \leq z \mathrel{\breve{\to}} (x < y) \leq z\,]$ says that $[\,<\,]^\circ$ and $[\,\leq\,]^\circ$ are left associative. I could just as well have stated the rule as $[\,x < y < z \mathrel{\breve{\to}} (x < y) < z\,]$ or $[\,x \leq y \leq z \mathrel{\breve{\to}} (x \leq y) \leq z\,]$, but I have chosen the form $[\,x < y \leq z \mathrel{\breve{\to}} (x < y) \leq z\,]$ to make the left hand side of the associativity rule identical to the left hand side of the term reduction rule.

To show you what the term reduction rule does, I now compute the value of $[\,2 < 5 \leq 8\,]$:

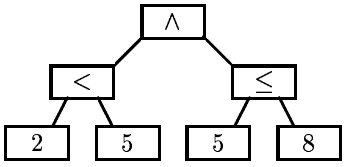


The first two steps above show how I use parse reductions to convert $[\,2 < 5 \leq 8\,]$ into a parse tree (each of the two steps of course involves several parse reductions). The parse tree reads



I use the associativity rule $[\,x < y \leq z \mathrel{\breve{\to}} (x < y) \leq z\,]$ to see that $[\,\leq\,]^\circ$ is the principal operator in the first step.

In the third step I perform a *term reduction*. In that step, I replace the left hand side of $\big[\,x < y \leq z \overset{\circ}{\to} x < y \wedge y \leq z\,\big]$ by the right hand side. This leads to the syntax tree of $[\,2 < 5 \leq 8\,]$ which reads:



$\big[\,x \overset{\circ}{\to} y\,\big]$ term reduction rule
x term reduces to y
term reduction

In the last three steps I perform graph reductions to compute the value. The value turns out to be $[\,\mathsf{T}\,]^{\cdot}$, so $[\,2 < 5 \leq 8\,]$ is true.

In words, $[\,2 < 5 \leq 8\,]$ reads "two is strongly less than five which in turn is weakly less than eight" or "two is strongly less than five and five is weakly less than eight".

$[\,2 < 5 \leq 8\,]^{\cdot}$ has the same syntax tree as $[\,2 < 5 \wedge 5 \leq 8\,]^{\cdot}$. In particular,

$$[\,2 < 5 \leq 8 \equiv 2 < 5 \wedge 5 \leq 8\,]^{\cdot}.$$

Recall that the term reduction rule reads

$$[\,\mathsf{x} < \mathsf{y} \leq \mathsf{z} \overset{\circ}{\rightarrow} \mathsf{x} < \mathsf{y} \wedge \mathsf{y} \leq \mathsf{z}\,].$$

$[\,<\,]^{\circ}$ occurs both at the left and the right hand side of $[\,\overset{\circ}{\rightarrow}\,]^{\circ}$. When you use the term reduction rule, you may replace $[\,<\,]^{\circ}$ by any operation that has the same priority as $[\,<\,]^{\circ}$. When you do so, you have to replace all occurrences of $[\,<\,]^{\circ}$ by the same operation. Furthermore, you may replace $[\,\leq\,]^{\circ}$ by any operation that has the same priority as $[\,\leq\,]^{\circ}$. Since $[\,<\,]^{\circ}$ and $[\,\leq\,]^{\circ}$ have the same priority, you may replace them by the same operations. Here you have some examples of use:

$$\begin{array}{lcl}
[\,\mathsf{x} = \mathsf{y} \neq 5 & \equiv & \mathsf{x} = \mathsf{y} \wedge \mathsf{y} \neq 5\,]^{\cdot} \\
[\,\mathsf{x} = \mathsf{y} = 6 & \equiv & \mathsf{x} = \mathsf{y} \wedge \mathsf{y} = 6\,]^{\cdot} \\
[\,\mathsf{x} \leq \mathsf{y} < 7 & \equiv & \mathsf{x} \leq \mathsf{y} \wedge \mathsf{y} < 7\,]^{\cdot} \\
[\,\mathsf{x} \leq \mathsf{y} \leq \mathsf{x} + 1 & \equiv & \mathsf{x} \leq \mathsf{y} \wedge \mathsf{y} \leq \mathsf{x} + 1\,]^{\cdot} \\
[\,\mathsf{x} < \mathsf{y} \wedge \mathsf{y} < \mathsf{u} < \mathsf{v} & \equiv & \mathsf{x} < \mathsf{y} \wedge (\mathsf{y} < \mathsf{u} \wedge \mathsf{u} < \mathsf{v})\,]^{\cdot}
\end{array}$$

Exercise 4.8.1 What are the values of the following:

(a) $[\,1 \neq 2 \neq 1\,]^{\cdot}$.

(b) $[\,(2 < 3 < 2) = (3 < 2 < 3)\,]^{\cdot}$.

4.9 Associativity of implication

I treat implication like I treat relations:

$$[\,\mathsf{x} \Rightarrow \mathsf{y} \Leftrightarrow \mathsf{z} \overset{\smile}{\rightarrow} (\mathsf{x} \Rightarrow \mathsf{y}) \Leftrightarrow \mathsf{z}\,], \text{ and}$$

$$[\,\mathsf{x} \Rightarrow \mathsf{y} \Leftrightarrow \mathsf{z} \overset{\circ}{\rightarrow} (\mathsf{x} \Rightarrow \mathsf{y}) \wedge (\mathsf{y} \Leftrightarrow \mathsf{z})\,].$$

As an example, these rules make Map interpret

$$[\,\mathsf{x} = 3 \Rightarrow \mathsf{x}^2 = 9 \Leftrightarrow \mathsf{x}^2 + 1 = 10\,]^{\cdot}$$

as

$$[\,(\mathsf{x} = 3 \Rightarrow \mathsf{x}^2 = 9) \wedge (\mathsf{x}^2 = 9 \Leftrightarrow \mathsf{x}^2 + 1 = 10)\,]^{\cdot}.$$

Exercise 4.9.1 What are the values of the following:

(a) $[\,1 = 1 \Rightarrow 1 = 2 \Rightarrow 2 = 2\,]^{\cdot}$.

(b) $[\,1 = 2 \Rightarrow 2 = 2 \Rightarrow 1 = 1\,]^{\cdot}$.

4.10 Reduction order in forming syntax trees

In general, to find the syntax tree of an expression, perform the following three steps:

(1) Perform *parse reductions* until no further parse reductions are possible. You need priority and associativity rules to perform these reductions. See Appendix A.2 for priority rules and Appendix A.3 for associativity rules. The result from this step is the *parse tree*.

(2) Perform *term reductions* until no further term reductions are possible. You need term reduction rules to perform these reductions. See Appendix A.3 for term reduction rules.

(3) Perform *macro reductions* until no further macro reductions are possible. You need macro definitions to perform these reductions. Use the index to locate macro definitions. The result from this step is the *syntax tree*.

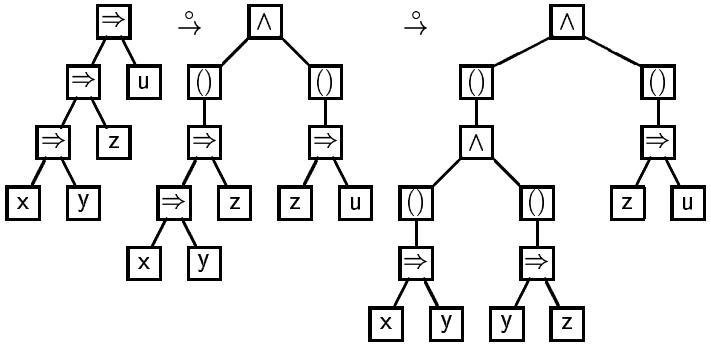
In Step 2, the order in which you perform term reductions is important. Here you have a more elaborate version of Step 2:

(2a) Apply term reduction to the root of $[\,\mathcal{A}\,]^{\cdot}$ until no further rules apply.

(2b) Apply term reduction to all subtrees of the root until no further rules apply.

(2c) Start over from (2a) until no further rules apply.

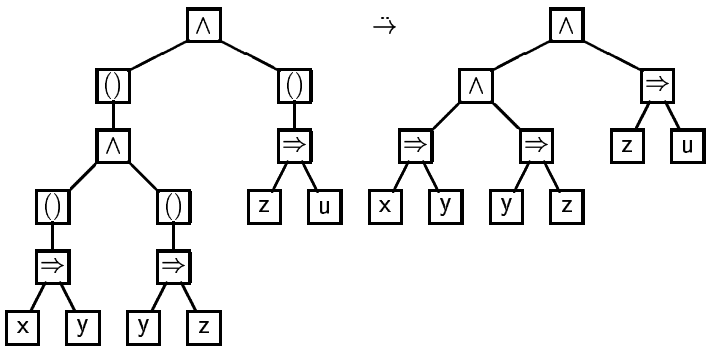
As an example, $[\,x \Rightarrow y \Rightarrow z \Rightarrow u\,]^{\cdot}$ first term reduces thus:



and then macro reduces thus:

---

parse reduction
parse tree
term reduction
macro reduction
syntax tree



Hence, $[\![\, x \Rightarrow y \Rightarrow z \Rightarrow u \,]\!]^{\cdot}$ and $[\![\, (x \Rightarrow y) \wedge (y \Rightarrow z) \wedge (z \Rightarrow u) \,]\!]^{\cdot}$ denote the same syntax tree. As you can see in the reduction above, the root is reduced first in accordance with the reduction order.

In the example above, I perform term reductions until no further term reductions are possible before I remove the parentheses. This is because parentheses are macros.

4.11 Commas

I will use the construct $[\![\, \boxed{x, y} \,]\!]^{\cdot}$ quite a lot, but the construct is peculiar in that it has no meaning of its own. It merely has a meaning in certain contexts. The following tells map that $[\![\, x, y \,]\!]^{\cdot}$ is a construct without a meaning of its own:

$$[\![\, \textbf{construct}\, x, y \,]\!].$$

A statement like $[\![\, \mathcal{A} \doteq \mathcal{B} \,]\!]^{\cdot}$ introduces a construct $[\![\, \mathcal{A} \,]\!]^{\cdot}$ and assigns a meaning to it. An $[\![\, \mathcal{A} \,]\!]^{\cdot}$ that is introduced by a definition of for $[\![\, \mathcal{A} \doteq \mathcal{B} \,]\!]^{\cdot}$ becomes an operator.

$[\![\, \boxed{\textbf{construct}\, \mathcal{A}} \,]\!]^{\cdot}$, in contrast, introduces the construct $[\![\, \mathcal{A} \,]\!]^{\cdot}$ but assigns no meaning. An $[\![\, \mathcal{A} \,]\!]^{\cdot}$ that is introduced by $[\![\, \textbf{construct}\, \mathcal{A} \,]\!]^{\cdot}$ becomes a directive. Map would protest against $[\![\, (2,3) \neq (3,2) \,]\!]^{\cdot}$ if you removed the dot since Map does not know how to compute $[\![\, (2,3) \,]\!]^{\cdot}$ and $[\![\, (3,2) \,]\!]^{\cdot}$.

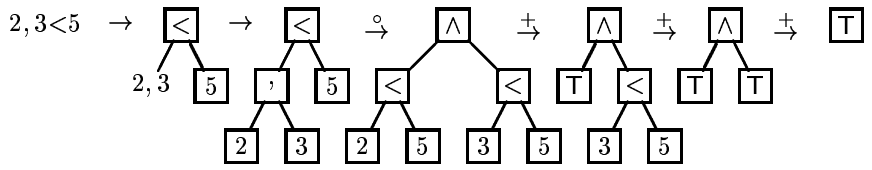
I make the comma right associative $[\![\, x, y, z \mathrel{\tilde{\rightarrow}} x, (y, z) \,]\!]^{\cdot}$ and give it priority above relations $[\![\, x + y \mathrel{\breve{>}} x, y \mathrel{\breve{>}} x = y \,]\!]^{\cdot}$.

Here you have a term reduction rule that assigns a meaning to a comma in the context of a relation:

$$[\![\, x, y < z \mathrel{\mathring{\rightarrow}} x < z \wedge y < z \,]\!].$$

As an example of use, I will compute the value of $[\![\, 2, 3 < 5 \,]\!]$:

$[\![\, x, y \,]\!]^{\cdot}$	x comma y
$[\![\, \textbf{construct}\, x \,]\!]^{\cdot}$	construct x



As you can see, $[\,2,3 < 5\,]^{\cdot}$ is true. $[\,2,3 < 5\,]^{\cdot}$ means "two and three are strongly less than five".

Here you have another term reduction rule:

$$[\,x < y, z \overset{\circ}{\to} x < y \wedge x < z\,].$$

As an example, $[\,3 < 5, 6 \equiv 3 < 5 \wedge 3 < 6\,]^{\cdot}$ so $[\,3 < 5, 6\,]^{\cdot}$ is true. $[\,3 < 5, 6\,]^{\cdot}$ means "three is strongly less than five and six".

Exercise 4.11.1 What is the syntax tree and the value of $[\,(2 < 3 < 2) < (2 < 3, 4)\,]^{\cdot}$

4.12 Term reduction priority

The term reduction rules above will confuse Map if you write a statement like $[\,2,3 < 4,5\,]^{\cdot}$. Which of the two rules should Map compute first? And what about $[\,2 < 3 < 4,5\,]^{\cdot}$? I have decided to use $[\,\breve{>}\,]^{\circ}$ and $[\,\breve{=}\,]^{\circ}$ to assign priorities to reduction rules:

$$\begin{array}{ll}[& x < y \leq z \overset{\circ}{\to} (x < y) \wedge (y \leq z) \breve{>} \\ & x, y < z \overset{\circ}{\to} (x < z) \wedge (y < z) \breve{>} \\ & x < y, z \overset{\circ}{\to} (x < y) \wedge (x < z)\,]\end{array}$$

If Map makes term reduction of the root of a syntax tree and if more than one term reduction rule applies, then Map chooses the one with highest priority. If more than one term reduction rule has maximal priority, then Map complains. Here you have some examples:

$$\begin{array}{lcl}[\,2,3 < 4,5 & \equiv & (2 < 4,5) \wedge (3 < 4,5) \\ & \equiv & (2 < 4 \wedge 2 < 5) \wedge (3 < 4 \wedge 3 < 5)\,]^{\cdot}, \\ [\,1 + 2 = 3 < 4,5 & \equiv & 1 + 2 = 3 \wedge (3 < 4,5) \\ & \equiv & 1 + 2 = 3 \wedge (3 < 4 \wedge 3 < 5)\,]^{\cdot}, \text{ and} \\ [\,2 < x, y < 5 & \equiv & (2 < x, y) \wedge (x, y < 5) \\ & \equiv & (2 < x \wedge 2 < y) \wedge (x < 5 \wedge y < 5)\,]^{\cdot}.\end{array}$$

Exercise 4.12.1 What are the values of the following:

(a) $[\,1,2,3 < 4,5,6\,]^{\cdot}$.

(b) $[\,1,3,5 < 2,4,6\,]^{\cdot}$.

(c) $[\,1,3,5 \neq 2,4,6\,]^{\cdot}$.

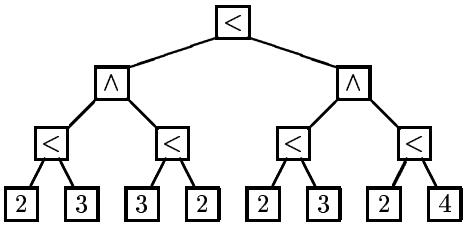
(d) $[\![\, 1,3,5 \neq 2,2,2 \,]\!]$.

(e) $[\![\, 1,2,3 \neq 2,4,6 \,]\!]$.

(f) $[\![\text{if}(1<2, 2<3, 3<4, 4<5) \,]\!]$.

4.13 Answers

Answer 4.2.1 (a) $[\![\, \mathsf{F} \,]\!]$. (b) $[\![\, 3 \,]\!]$. (c) $[\![\, \bot \,]\!]$.
Answer 4.4.1 (a) $[\![\, \infty \,]\!]$. (b) $[\![\, \bullet \,]\!]$.
Answer 4.5.1 (a) $[\![\, \mathsf{T} \,]\!]$. (b) $[\![\, \mathsf{F} \,]\!]$.
Answer 4.6.1 (a) $[\![\, \mathsf{F} \,]\!]$. (b) $[\![\, \mathsf{T} \,]\!]$.
Answer 4.7.1 (a) $[\![\, \mathsf{T} \,]\!]$. (b) $[\![\, \mathsf{T} \,]\!]$.
Answer 4.8.1 (a) $[\![\, \mathsf{T} \,]\!]$. (b) $[\![\, \mathsf{T} \,]\!]$.
Answer 4.9.1 (a) $[\![\, \mathsf{F} \,]\!]$. (b) $[\![\, \mathsf{T} \,]\!]$.
Answer 4.11.1 The value is $[\![\, \mathsf{T} \,]\!]$. The syntax tree reads:



Answer 4.12.1 (a) $[\![\, \mathsf{T} \,]\!]$. (b) $[\![\, \mathsf{F} \,]\!]$. (c) $[\![\, \mathsf{T} \,]\!]$. (d) $[\![\, \mathsf{T} \,]\!]$. (e) $[\![\, \mathsf{F} \,]\!]$. (f) The expresion has three possible interpretations:

- $[\![\text{if}((1<2, 2<3), 3<4, 4<5) \,]\!]$.
- $[\![\text{if}(1<2, (2<3, 3<4), 4<5) \,]\!]$.
- $[\![\text{if}(1<2, 2<3, (3<4, 4<5)) \,]\!]$.

An expression that has more than one interpretation makes Map protest. Such an expression is said to be *ambiguous* and it has no value.

---

ambiguous

Chapter 5

Pairs

5.1 Coordinates

I wrote these lines about [56]⁻ degrees North of equator and about [12]⁻ degrees East of Greenwich. The ordinates [56]⁻ and [12]⁻ allow you to spot the location if you wish. You need both of the ordinates to spot the location, and for that reason such ordinates are known as *coordinates*.

In general, you can specify a location by two numbers. In many situations, however, it is inconvenient to represent a single entity like a location by several mathematical entities such as two numbers. For that reason there is a need for a way to aggregate several mathematical entities into a single mathematical entity.

In this chapter I introduce the notion of a *pair* and the notion of a *list*. A pair is something that aggregates two mathematical entities into a single mathematical entity. A list is something that aggregates several mathematical entities into a single entity.

Here you have a pair of numbers:

[56 :: 12]⁻.

The pair consists of two numbers, [56]⁻ and [12]⁻, which are stated in a particular order, namely [56]⁻ first and [12]⁻ last. In general, if [x]⁻ and [y]⁻ are mathematical entities, then

$$\left[\boxed{x :: y}\right]^{-}$$

is the pair that consists of [x]⁻ and [y]⁻ in that order.

You can represent locations by pairs of numbers. As an example, you can represent a location by a pair whose first element is the northern lattitude of

---

coordinates
pair
list
[x :: y]⁻ x pair y

the location and the second is the eastern longitude. With this convention, I wrote these lines at $[\![56 :: 12]\!]$.

$[\![56 :: 12]\!]$ and $[\![12 :: 56]\!]$ are two different pairs. They differ in the order of the numbers.

A pair may have the same number as its first and second component. $[\![10 :: 10]\!]$ is an example of such a pair.

$[\![x :: y]\!]$ is a binary infix operator. It has priority between $[\![x + y]\!]$ and $[\![x, y]\!]$:

$$[\![x + y \mathrel{\breve{>}} x :: y \mathrel{\breve{>}} x, y]\!].$$

$[\![x :: y]\!]$ is right associative:

$$[\![x :: y :: z \mathrel{\breve{\rightarrow}} x :: (y :: z)]\!].$$

5.2 Equality of pairs

$[\![x :: y \equiv u :: v]\!]$ holds if $[\![x \equiv u]\!]$ holds and $[\![y \equiv v]\!]$ holds. $[\![x :: y \equiv u :: v]\!]$ fails if $[\![x \equiv u]\!]$ fails or $[\![y \equiv v]\!]$ fails or both fail. As an example,

$[\![1 + 2 :: 5 \equiv 3 :: 5]\!]$ holds

because $[\![1 + 2 \equiv 3]\!]$ holds and $[\![5 \equiv 5]\!]$ holds.

$[\![1 + 2 :: 5 \equiv 4 :: 5]\!]$ fails

because $[\![1 + 2 \equiv 4]\!]$ fails. Furthermore,

$[\![4 :: 5 \equiv 5 :: 4]\!]$ fails

because $[\![4 \equiv 5]\!]$ and $[\![5 \equiv 4]\!]$ fail.

$[\![1 :: 2 + 3 :: 4 \equiv 1 :: 5 :: 4]\!]$ holds

because $[\![1 \equiv 1]\!]$ and $[\![2 + 3 :: 4 \equiv 5 :: 4]\!]$ hold. The latter holds because $[\![2 + 3 \equiv 5]\!]$ and $[\![4 \equiv 4]\!]$ hold.

5.3 Head and tail

The construct $[\![x :: y]\!]$ does not really do anything to $[\![x]\!]$ and $[\![y]\!]$. It just puts $[\![x]\!]$ and $[\![y]\!]$ together in such a way that you can take them apart again. You can take a pair apart by means to the constructs $\left[\!\left[\boxed{z \text{ head}} \right]\!\right]$ and $\left[\!\left[\boxed{z \text{ tail}} \right]\!\right]$. If $[\![z]\!]$ is a pair, then $[\![z \text{ head}]\!]$ equals the first element of the pair and $[\![z \text{ tail}]\!]$

$[\![x \text{ head}]\!]$ x head
$[\![x \text{ tail}]\!]$ x tail

equals the second. In other words, if $[\![x]\!]$ and $[\![y]\!]$ are arbitrary mathematical entities then

$[\![(x :: y)\ \text{head} \equiv x]\!]$, and

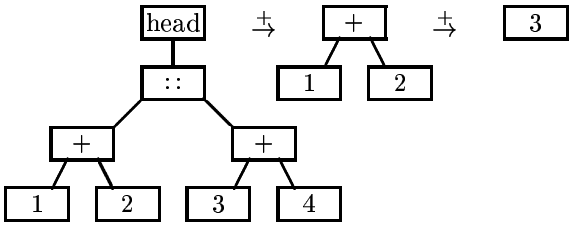
$[\![(x :: y)\ \text{tail} \equiv y]\!]$.

Map uses the following reduction rules to compute the head and tail of a pair:

$[\![(x :: y)\ \text{head} \xrightarrow{+} x]\!]$, and

$[\![(x :: y)\ \text{tail} \xrightarrow{+} y]\!]$.

Here you can see how Map computes $[\![(1 + 2 :: 3 + 4)\ \text{head}]\!]$:



As you can see, Map is *lazy* in the sense that it does not compute $[\![3 + 4]\!]$ (c.f. Section 2.10). Map merely computes what it is *forced* to compute.

Exercise 5.3.1 Draw parse trees of the following and compute their values:

(a) $[\![((1 :: 2) :: (3 :: 4))\ \text{head}]\!]$.

(b) $[\![((1 :: 2) :: (3 :: 4))\ \text{head tail}]\!]$.

(c) $[\![((1 :: 2) :: (3 :: 4))\ \text{tail tail}]\!]$.

5.4 Strictness of head and tail

Computation of $[\![(-1)!\ \text{head}]\!]$ proceeds thus:

$$\begin{aligned} [\![(-1)!\ \text{head} &\xrightarrow{+} ((-1) \cdot (-2)!)\ \text{head} \\ &\xrightarrow{+} ((-1) \cdot (-2) \cdot (-3)!)\ \text{head} \\ &\xrightarrow{+} \cdots]\!]^{\circ} \end{aligned}$$

In other words, $[\![\bot\ \text{head} \equiv \bot]\!]$, so $[\![x\ \text{head}]\!]$ is strict. Likewise, $[\![x\ \text{tail}]\!]$ is strict.

If $[\![x]\!]$ is a truth value, a decimal fraction or an exception, then I make the choice that

$[\![x\ \text{head} \equiv x]\!]$, and

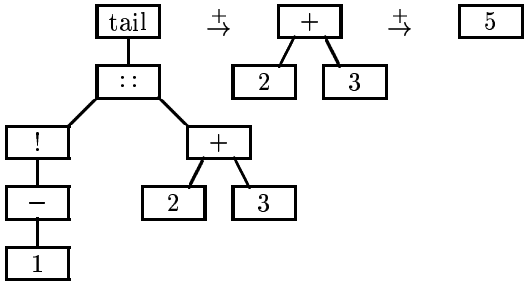
$[\![x\ \text{tail} \equiv x]\!]$.

As examples, $[\![\mathsf{T}\ \text{head} \equiv \mathsf{T}]\!]$ and $[\![2.0\mathsf{F}\ \text{head} \equiv 2.0\mathsf{F}]\!]$.

lazy
force

5.5 Laziness of $[\![x :: y]\!]$

Here you can see how Map computes $[\![((-1)! :: 2 + 3)\ \text{tail}]\!]$:



As you can see, Map can compute $[\![((-1)! :: 2 + 3)\ \text{tail}]\!]$ in finite time even though the expression contains $[\![(-1)!]\!]$ which takes an infinite amount of time to compute. As you can see, the laziness of Map saves an infinite amount of time here.

The last example above shows that

$$[\![(x :: y)\ \text{tail} \equiv y]\!]$$

holds even when $[\![x]\!]$ is non-informative.

I will now consider the following question: Is $[\![\bot :: 5]\!]$ equal to $[\![\bot]\!]$? If the answer were yes, then you would have $[\![(\bot :: 5)\ \text{tail} \equiv \bot\ \text{tail} \equiv \bot]\!]$. But $[\![(\bot :: 5)\ \text{tail}]\!]$ is equal to $[\![5]\!]$, so the answer cannot be yes. Therefore, the answer must be: no, $[\![\bot :: 5]\!]$ differs from $[\![\bot]\!]$.

You now have an example of a $[\![y]\!]$ for which $[\![\bot :: y]\!]$ differs from $[\![\bot]\!]$, namely $[\![y \equiv 5]\!]$. Therefore, $[\![x :: y]\!]$ is non-strict in $[\![x]\!]$.

If you replace $[\![z\ \text{tail}]\!]$ by $[\![z\ \text{head}]\!]$ and interchange $[\![\bot]\!]$ and $[\![5]\!]$ in the argument above you will see that $[\![x :: y]\!]$ is also non-strict in $[\![y]\!]$.

$[\![x :: y]\!]$ is non-strict because it is non-strict in at least one of its arguments. $[\![x :: y]\!]$ is unusual in that it is non-strict in all of its arguments.

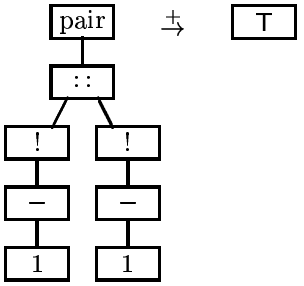
5.6 Pairhood

$\left[\!\left[\boxed{z\ \text{pair}} \right]\!\right]$ can test for "pairhood". $[\![z\ \text{pair}]\!]$ equals $[\![T]\!]$ if $[\![z]\!]$ is a pair and equals $[\![F]\!]$ if $[\![z]\!]$ is a truth value, a decimal fraction, or an exception. $[\![z\ \text{pair}]\!]$ is strict so $[\![\bot\ \text{pair} \equiv \bot]\!]$. Here you have some reduction rules that Map uses:

$$\begin{array}{rcl}
[\![(x :: y)\ \text{pair} & \xrightarrow{+} & T]\!] \\
[\![T\ \text{pair} & \xrightarrow{+} & F]\!] \\
[\![F\ \text{pair} & \xrightarrow{+} & F]\!] \\
[\![\bullet\ \text{pair} & \xrightarrow{+} & F]\!] \\
[\![x\ \text{pair} & \xrightarrow{+} & F]\!] \quad \text{if } [\![x]\!] \text{ is a decimal fraction}
\end{array}$$

$$\frac{}{[\![x\ \text{pair}]\!]\quad x\ \text{pair}}$$

As an example, Map computes $[\,((-1)!::(-1)!)\ \text{pair}\,]^{\cdot}$ thus:



In other words:

$$[\,(\bot :: \bot)\ \text{pair} \equiv \mathsf{T}\,]^{\cdot}.$$

I will now consider the following question: Is $[\,\bot :: \bot\,]^{\cdot}$ equal to $[\,\bot\,]^{\cdot}$? If the answer were yes, then you would have $[\,(\bot :: \bot)\ \text{pair} \equiv \bot\ \text{pair} \equiv \bot\,]^{\cdot}$. But $[\,(\bot :: \bot)\ \text{pair}\,]^{\cdot}$ equals $[\,\mathsf{T}\,]^{\cdot}$, so the answer cannot be yes. Therefore, the answer must be: no, $[\,\bot :: \bot\,]^{\cdot}$ differs from $[\,\bot\,]^{\cdot}$.

How can $[\,\bot :: \bot\,]^{\cdot}$ contain information when it is just no information aggregated with no information? The answer is that $[\,\bot :: \bot\,]^{\cdot}$ contains the information that it is a pair. $[\,\bot :: \bot\,]^{\cdot}$ contains no more information than this. In conclusion, $[\,\bot :: \bot\,]^{\cdot}$ contains very little information, but it contains more than nothing.

The opposite of a "pair" is known as an *atom*:

$$\big[\,\boxed{\text{x atom}} \doteq \neg(\text{x pair})\,\big].$$

As examples, truth values, decimal fractions and exceptions are atoms. $[\,\bot\,]^{\cdot}$ is neither a pair nor an atom.

5.7 Lists

I wrote these lines about $[\,56\,]^{\cdot}$ degrees North of equator, about $[\,12\,]^{\cdot}$ degrees East of Greenwich and about $[\,20\,]^{\cdot}$ meters above sea level. These three coordinates fix the location in three dimensions. To aggregate these coordinates into a single mathematical entity, I need something that can aggregate three entities, and a pair can merely aggregate two.

I will use the notion of a *list* to aggregate several mathematical entities into one. Here you have an example of a list:

$$[\,\langle 56, 12, 20\rangle\,]^{\cdot}.$$

	atom
$[\,\text{x atom}\,]^{\cdot}$	x atom
	list

The list has three elements. The first element is $[\,56\,]^{\cdot}$, the second element is $[\,12\,]^{\cdot}$, and the third is $[\,20\,]^{\cdot}$.

You can represent locations by lists of numbers. As an example, you can represent a location by a three element list whose first element is the Northern lattitude, the second element is the Eastern longitude, and the third element is the altitude above sea level. With this convention, I wrote these lines at $[\,\langle 56, 12, 20\rangle\,]^{\cdot}$.

A list can have any number of elements. Here you have a list with $[\,8\,]^{\cdot}$ elements:

$$[\,\langle 1, 2, 3, 4, 5, 6, 7, 8\rangle\,]^{\cdot}.$$

Here you have a list with two elements:

$$[\,\langle 1, 2\rangle\,]^{\cdot}.$$

A list with two elements is not the same as a pair of two elements:

$$[\,\langle 1, 2\rangle \equiv 1 :: 2\,]^{\cdot} \text{ fails.}$$

I explain the relation between lists and pairs in a moment. Here you have a list of one element:

$$[\,\langle 1\rangle\,]^{\cdot}.$$

A list with one element is not the same as the element:

$$[\,\langle 1\rangle \equiv 1\,]^{\cdot} \text{ fails.}$$

Here you have a list with zero elements:

$$\Big[\boxed{\langle\,\rangle}\Big]^{\cdot}.$$

There is only one list that has zero elements, and that list is known as the *empty list*.

Two lists are equal if they have the same number of elements and the elements in the list are pairwise equal and stated in the same order. Here you have some examples:

$$[\,\langle 1, 2, 3\rangle \equiv \langle 1, 2, 3, 3\rangle\,]^{\cdot} \text{ fails,}$$

$$[\,\langle 1, 2, 3\rangle \equiv \langle 1, 2, 4\rangle\,]^{\cdot} \text{ fails,}$$

$$[\,\langle 1, 2, 3\rangle \equiv \langle 1, 3, 2\rangle\,]^{\cdot} \text{ fails, and}$$

$$[\,\langle 1, 2, 3\rangle \equiv \langle 1, 2, 3\rangle\,]^{\cdot} \text{ holds.}$$

$[\,\langle\,\rangle\,]^{\cdot}$ empty list
empty list

5.8 Representation of lists

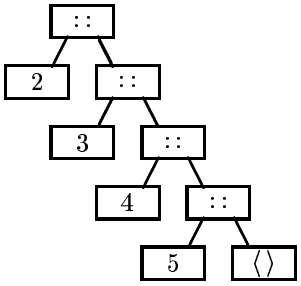
John McCarthy (American computer scientist) has defined a computer programming language called *Lisp* [3]. Lisp stands for "list processor" and is a programming language in which lists play a central role. One of the properties of Lisp is that the pair operator $[\,x :: y\,]^{\cdot}$ and the empty list $[\,\langle\,\rangle\,]^{\cdot}$ are used to represent lists of arbitrary length. As an example, the list

$$[\,\langle 2,3,4,5\rangle\,]^{\cdot}$$

is considered to be shorthand for

$$[\,2 :: 3 :: 4 :: 5 :: \langle\,\rangle\,]^{\cdot}.$$

In other words, the syntax tree of $[\,\langle 2,3,4,5\rangle\,]^{\cdot}$ reads

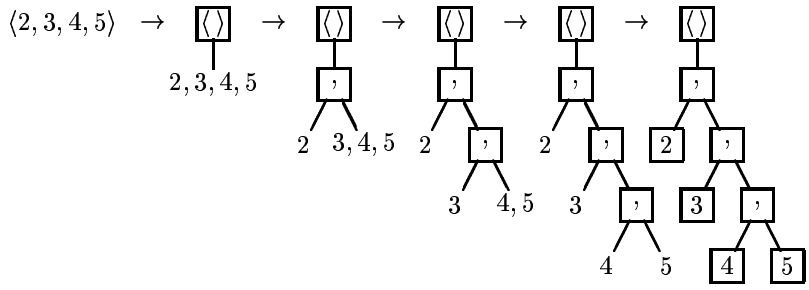


Here you have the definitions that make Map interpret $[\,\langle 2,3,4,5\rangle\,]^{\cdot}$ as $[\,2 :: 3 :: 4 :: 5 :: \langle\,\rangle\,]^{\cdot}$:

$$\big[\,\langle x,y\rangle \overset{\circ}{\to} x :: \langle y\rangle\,\big], \text{ and}$$

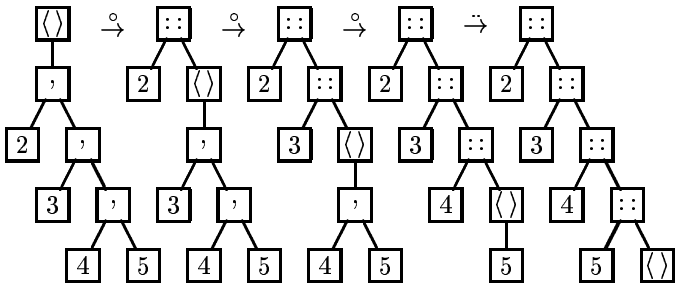
$$\Big[\,\boxed{\langle x\rangle} \doteq x :: \langle\,\rangle\,\Big].$$

Here you see how Map constructs the syntax tree of $[\,\langle 2,3,4,5\rangle\,]^{\cdot}$. Map first constructs the parse tree:



	Lisp
$[\,\langle x\rangle\,]^{\cdot}$	tuple x end

Then Map performs term reductions until no further term reductions are possible and then performs macro reductions until no further macro reductions are possible:



The empty list is not a pair. Rather, the empty list is an atom:

$$\begin{array}{lcl} [\,\langle\rangle\ \mathsf{pair} & \equiv & \mathsf{F}\,] \\ [\,\langle\rangle\ \mathsf{atom} & \equiv & \mathsf{T}\,] \\ [\,\langle\rangle\ \mathsf{head} & \equiv & \langle\rangle\,] \\ [\,\langle\rangle\ \mathsf{tail} & \equiv & \langle\rangle\,] \end{array}$$

My notation differs significantly from that of McCarthy [3], but the fundamental idea is the same: Introduce the notion of pairs $[\,\mathsf{x} :: \mathsf{y}\,]$ and introduce the notion of the empty list $[\,\langle\rangle\,]$. Then use these two notions to introduce lists of arbitrary length.

Exercise 5.8.1 Draw the syntax trees of the following:

(a) $[\,\langle 1,2,3,4,5\rangle\,]$.

(b) $[\,\langle\langle 1,2\rangle,\langle 3,4\rangle\rangle\,]$.

(c) $[\,\langle 1,(2,3)\rangle\,]$.

5.9 Pairs and recursion

Here you have a definition where the construct $[\,\mathsf{x} :: \mathsf{y}\,]$ occurs on the right hand side:

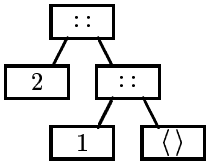
$$\left[\boxed{\mathrm{nlist}(\mathsf{n})} \doteq \mathsf{n} = 0 \left\{ \begin{array}{l} \langle\rangle \\ \mathsf{n} :: \mathrm{nlist}(\mathsf{n}-1) \end{array} \right.\right].$$

Here you have a computation of $[\,\mathrm{nlist}(2)\,]$:

$$\begin{array}{lcl} \mathrm{nlist}(2) & \stackrel{+}{\rightarrow} & 2 :: \mathrm{nlist}(1) \\ & \stackrel{+}{\rightarrow} & 2 :: 1 :: \mathrm{nlist}(0) \\ & \stackrel{+}{\rightarrow} & 2 :: 1 :: \langle\rangle \end{array}$$

$[\,\mathrm{nlist}(\mathsf{x})\,]$ n list of x end

Hence, $[\![\ \mathrm{nlist}(2)\ ]\!]^{\cdot}$ looks thus:



Exercise 5.9.1 Draw $[\![\ \mathrm{nlist}(4)\ ]\!]^{\cdot}$. Express $[\![\ \mathrm{nlist}(4)\ ]\!]^{\cdot}$ as a list of form $[\![\ \langle \cdots \rangle\ ]\!]^{\circ}$.

Exercise 5.9.2 I define

$$\left[\!\left[\ \boxed{\mathrm{nnlist}(\mathsf{n})} \doteq \mathsf{n} = 0 \left\{ \begin{array}{l} \langle\,\rangle \\ \mathsf{T} :: (\mathsf{F} :: \mathrm{nnlist}(\mathsf{n}-1)) \end{array} \right.\ \right]\!\right].$$

Draw $[\![\ \mathrm{nnlist}(2)\ ]\!]^{\cdot}$. Express $[\![\ \mathrm{nnlist}(4)\ ]\!]^{\cdot}$ as a list of form $[\![\ \langle \cdots \rangle\ ]\!]^{\circ}$.

5.10 Head and tail of lists

You have that $[\![\ \mathsf{x} :: \mathsf{y}\ ]\!]^{\cdot}$ is right associative, so

$$[\![\ 1 :: 2 :: 3 :: 4 :: 5 :: \langle\,\rangle\ ]\!]^{\cdot}$$

means

$$[\![\ 1 :: (2 :: (3 :: (4 :: (5 :: \langle\,\rangle))))\ ]\!]^{\cdot}.$$

Therefore you have

$$[\![\ (1 :: 2 :: 3 :: 4 :: 5 :: \langle\,\rangle)\ \mathrm{head} \equiv 1\ ]\!]^{\cdot}, \text{ and}$$

$$[\![\ (1 :: 2 :: 3 :: 4 :: 5 :: \langle\,\rangle)\ \mathrm{tail} \equiv 2 :: 3 :: 4 :: 5 :: \langle\,\rangle\ ]\!]^{\cdot}.$$

Or, using list notation:

$$[\![\ \langle 1,2,3,4,5\rangle\ \mathrm{head} \equiv 1\ ]\!]^{\cdot}, \text{ and}$$

$$[\![\ \langle 1,2,3,4,5\rangle\ \mathrm{tail} \equiv \langle 2,3,4,5\rangle\ ]\!]^{\cdot}.$$

The first element of a list is known as the *head of the list*. If $[\![\ \mathsf{x}\ ]\!]^{\cdot}$ is a list then $[\![\ \mathsf{x}\ \mathrm{head}\ ]\!]^{\cdot}$ denotes the head of the list. If you chop off the head of the list, then the

---

$[\![\ \mathrm{nnlist}(\mathsf{x})\ ]\!]^{\cdot}$ n n list of x end
head of list

remains are known as the *tail of the list*. If $[\,\mathsf{x}\,]^{\cdot}$ is a list then $[\,\mathsf{x}\ \text{tail}\,]^{\cdot}$ denotes the tail of the list. Here you have some further examples:

$$\begin{array}{lcr}
[\,\langle 2,3,4,5\rangle\ \text{head}\,]^{\cdot} & \equiv & 2\,]^{\cdot} \\
[\,\langle 2,3,4,5\rangle\ \text{tail}\,]^{\cdot} & \equiv & \langle 3,4,5\rangle\,]^{\cdot} \\
[\,\langle 3,4,5\rangle\ \text{head}\,]^{\cdot} & \equiv & 3\,]^{\cdot} \\
[\,\langle 3,4,5\rangle\ \text{tail}\,]^{\cdot} & \equiv & \langle 4,5\rangle\,]^{\cdot} \\
[\,\langle 4,5\rangle\ \text{head}\,]^{\cdot} & \equiv & 4\,]^{\cdot} \\
[\,\langle 4,5\rangle\ \text{tail}\,]^{\cdot} & \equiv & \langle 5\rangle\,]^{\cdot} \\
[\,\langle 5\rangle\ \text{head}\,]^{\cdot} & \equiv & 5\,]^{\cdot} \\
[\,\langle 5\rangle\ \text{tail}\,]^{\cdot} & \equiv & \langle\,\rangle\,]^{\cdot}
\end{array}$$

Here you have some examples of consecutive applications of $[\,\mathsf{x}\ \text{head}\,]^{\cdot}$ and $[\,\mathsf{x}\ \text{tail}\,]^{\cdot}$:

$$\begin{array}{lcr}
[\,\langle 1,2,3,4,5\rangle\ \text{tail}\,]^{\cdot} & \equiv & \langle 2,3,4,5\rangle\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{tail tail}\,]^{\cdot} & \equiv & \langle 3,4,5\rangle\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{tail tail tail}\,]^{\cdot} & \equiv & \langle 4,5\rangle\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{tail tail tail tail}\,]^{\cdot} & \equiv & \langle 5\rangle\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{tail tail tail tail tail}\,]^{\cdot} & \equiv & \langle\,\rangle\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{head}\,]^{\cdot} & \equiv & 1\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{tail head}\,]^{\cdot} & \equiv & 2\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{tail tail head}\,]^{\cdot} & \equiv & 3\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{tail tail tail head}\,]^{\cdot} & \equiv & 4\,]^{\cdot} \\
[\,\langle 1,2,3,4,5\rangle\ \text{tail tail tail tail head}\,]^{\cdot} & \equiv & 5\,]^{\cdot}
\end{array}$$

Exercise 5.10.1 I define

$$\left[\,\boxed{\text{minlist}(\mathsf{x})} \doteq \mathsf{x}\ \text{atom}\begin{cases} +\infty \\ \min(\mathsf{x}\ \text{head}, \text{minlist}(\mathsf{x}\ \text{tail})) \end{cases}\right].$$

Compute (a) $[\,\text{minlist}(\langle 5,2,8\rangle)\,]^{\cdot}$ and (b) $[\,\text{minlist}(\langle\,\rangle)\,]^{\cdot}$.

5.11 Infinite lists

Take a look at this definition:

$$\left[\,\boxed{\infty\text{-list}(\mathsf{n})} \doteq \mathsf{n} :: \infty\text{-list}(\mathsf{n}+1)\,\right].$$

Map may compute $[\,\infty\text{-list}(1)\ \text{head}\,]^{\cdot}$ thus:

$$\begin{array}{lcl}
[\,\infty\text{-list}(1)\ \text{head} & \xrightarrow{+} & (1 :: \infty\text{-list}(2))\ \text{head} \\
 & \xrightarrow{+} & 1\,]^{\cdot}
\end{array}$$

	tail of list
$[\,\text{minlist}(\mathsf{x})\,]^{\cdot}$	minimum of list x end
$[\,\infty\text{-list}(\mathsf{x})\,]^{\cdot}$	ascension list of x end

Hence, you have

$$\lceil\, \infty\text{-list}(1)\ \mathsf{head} \equiv 1 \,\rceil.$$

In other words, $\lceil\, \infty\text{-list}(1) \,\rceil$ is a list whose first element is $\lceil\, 1 \,\rceil$.

Map may compute $\lceil\, \infty\text{-list}(1)\ \mathsf{tail}\ \mathsf{head} \,\rceil$ thus:

$$\begin{array}{lcl} \lceil\, \infty\text{-list}(1)\ \mathsf{tail}\ \mathsf{head} & \stackrel{+}{\to} & (1 :: \infty\text{-list}(2))\ \mathsf{tail}\ \mathsf{head} \\ & \stackrel{+}{\to} & \infty\text{-list}(2)\ \mathsf{head} \\ & \stackrel{+}{\to} & (2 :: \infty\text{-list}(3))\ \mathsf{head} \\ & \stackrel{+}{\to} & 2 \,\rceil \end{array}$$

Hence, you have

$$\lceil\, \infty\text{-list}(1)\ \mathsf{tail}\ \mathsf{head} \equiv 2 \,\rceil.$$

In other words, $\lceil\, \infty\text{-list}(1) \,\rceil$ is a list whose second element is $\lceil\, 2 \,\rceil$. Here you have some further properties of $\lceil\, \infty\text{-list}(1) \,\rceil$:

$$\lceil\, \infty\text{-list}(1)\ \mathsf{tail}\ \mathsf{tail}\ \mathsf{head} \equiv 3 \,\rceil,$$

$$\lceil\, \infty\text{-list}(1)\ \mathsf{tail}\ \mathsf{tail}\ \mathsf{tail}\ \mathsf{head} \equiv 4 \,\rceil, \text{ and}$$

$$\lceil\, \infty\text{-list}(1)\ \mathsf{tail}\ \mathsf{tail}\ \mathsf{tail}\ \mathsf{tail}\ \mathsf{head} \equiv 5 \,\rceil.$$

In other words, the third, fourth, and fifth element of $\lceil\, \infty\text{-list}(1) \,\rceil$ are $\lceil\, 3 \,\rceil$, $\lceil\, 4 \,\rceil$, and $\lceil\, 5 \,\rceil$, respectively. In general, $\lceil\, \infty\text{-list}(1) \,\rceil$ is a list whose $\lceil\, \mathsf{n} \,\rceil$'th element equals $\lceil\, \mathsf{n} \,\rceil$ for all positive integers $\lceil\, \mathsf{n} \,\rceil$.

As you can see, $\lceil\, \infty\text{-list}(1) \,\rceil$ behaves as if it were a list with infinitely many elements. A computer program like Map cannot compute an infinite list in finite time. Furthermore, a computer program like Map merely has access to a finite amount of computer memory, and therefore Map cannot store an infinite list in its memory. Nevertheless, the laziness of Map allows Map to pretend that it can compute with infinitely long lists like $\lceil\, \infty\text{-list}(1) \,\rceil$. At any time, Map can merely keep a finite portion of $\lceil\, \infty\text{-list}(1) \,\rceil$ in its memory but, in principle, Map can compute any portion of the list. In this sense, $\lceil\, \infty\text{-list}(1) \,\rceil$ is a *potentially infinite list*: A computer can merely keep a finite portion of it in memory at any one time, but there is no limit in principle on how large this portion can be.

Exercise 5.11.1 **(a)** Draw the value of $\lceil\, \infty\text{-list}(1) \,\rceil$.

(b) Compute $\lceil\, \infty\text{-list}(0)\ \mathsf{tail}\ \mathsf{tail}\ \mathsf{head} \,\rceil$.

(c) Does $\lceil\, \infty\text{-list}(1) \equiv \bot \,\rceil$ hold or fail?

(d) Compute $\lceil\, \mathrm{minlist}(\infty\text{-list}(1)) \,\rceil$.

---

potentially infinite list

5.12 Operations and predicates on pairs

If $[\![x]\!]$ is a truth value, a decimal fraction, an exception, a pair or the empty list and if $[\![y]\!]$ is a pair or the empty list, then

$$[\![x + y \equiv \bullet]\!], \text{ and}$$

$$[\![y + x \equiv \bullet]\!].$$

Similar rules hold for $[\![x - y]\!]$, $[\![x \cdot y]\!]$, $[\![x \mathbin{//} y]\!]$, $[\![x \mathbin{\%} y]\!]$, $[\![x < y]\!]$, $[\![x \leq y]\!]$, $[\![x > y]\!]$, and $[\![x \geq y]\!]$.

If $[\![x]\!]$ is a truth value or a decimal fraction and $[\![y]\!]$ is a pair or the empty list, then

$$[\![x = y \equiv \mathsf{F}]\!], \text{ and}$$

$$[\![y = x \equiv \mathsf{F}]\!].$$

In other words, $[\![x = y]\!]$ considers pairs and the empty list to be different from truth values and decimal fractions. Furthermore, if $[\![y]\!]$ is a pair or the empty list, you have

$$[\![\bullet = y \equiv \bullet]\!], \text{ and}$$

$$[\![y = \bullet \equiv \bullet]\!].$$

In other words, $[\![x = y]\!]$ refuses to compare exceptions. Finally, you have

$$[\![x :: y = u :: v \equiv x = u \wedge y = v]\!],$$

$$[\![x :: y = \langle\rangle \equiv \mathsf{F}]\!],$$

$$[\![\langle\rangle = u :: v \equiv \mathsf{F}]\!], \text{ and}$$

$$[\![\langle\rangle = \langle\rangle \equiv \mathsf{T}]\!].$$

Exercise 5.12.1 What are the values of the following:

(a) $[\![\langle 1, 2\rangle = \langle 1, 2\rangle]\!]$.

(b) $[\![\langle 1, 2\rangle = \langle 1, 3\rangle]\!]$.

(c) $[\![\langle 1, 2\rangle = \langle 1, 2, 3\rangle]\!]$.

(d) $[\![\langle 1, 2\rangle = \langle 1, \mathsf{T}\rangle]\!]$.

(e) $[\![\langle 1, 2\rangle = \langle 1, \bullet\rangle]\!]$.

(f) $[\![\langle 1, 2\rangle = \langle 1, \bot\rangle]\!]$.

(g) $[\![\langle 1, 2, 3\rangle = \langle 4, \bullet, \bot\rangle]\!]$.

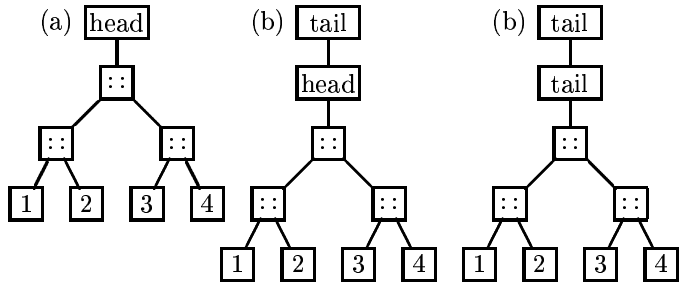
5.13 Answers

Answer 5.3.1

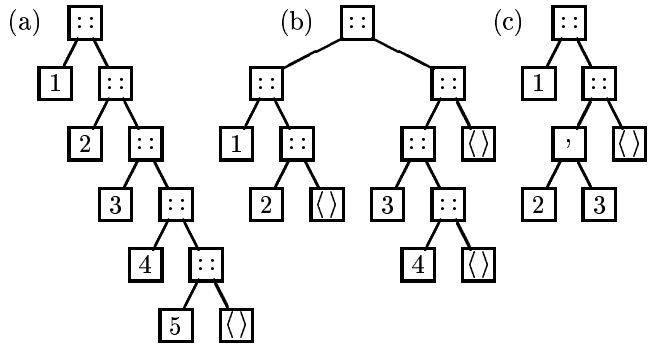
(a) $[\![\, 1 :: 2 \,]\!]$.

(b) $[\![\, 2 \,]\!]$.

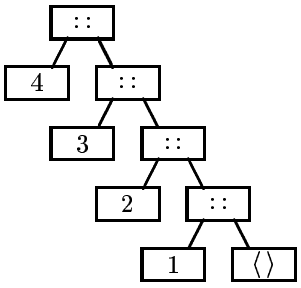
(c) $[\![\, 4 \,]\!]$.



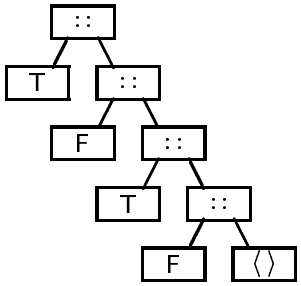
Answer 5.8.1



Answer 5.9.1 $[\![\, \langle 4, 3, 2, 1 \rangle \,]\!]$



Answer 5.9.2 $[\![\, \langle \mathsf{T}, \mathsf{F}, \mathsf{T}, \mathsf{F}, \mathsf{T}, \mathsf{F}, \mathsf{T}, \mathsf{F} \rangle \,]\!]$.



Answer 5.10.1 (a) $[\,2\,]^{\cdot}$ and (b) $[\,+\infty\,]^{\cdot}$.

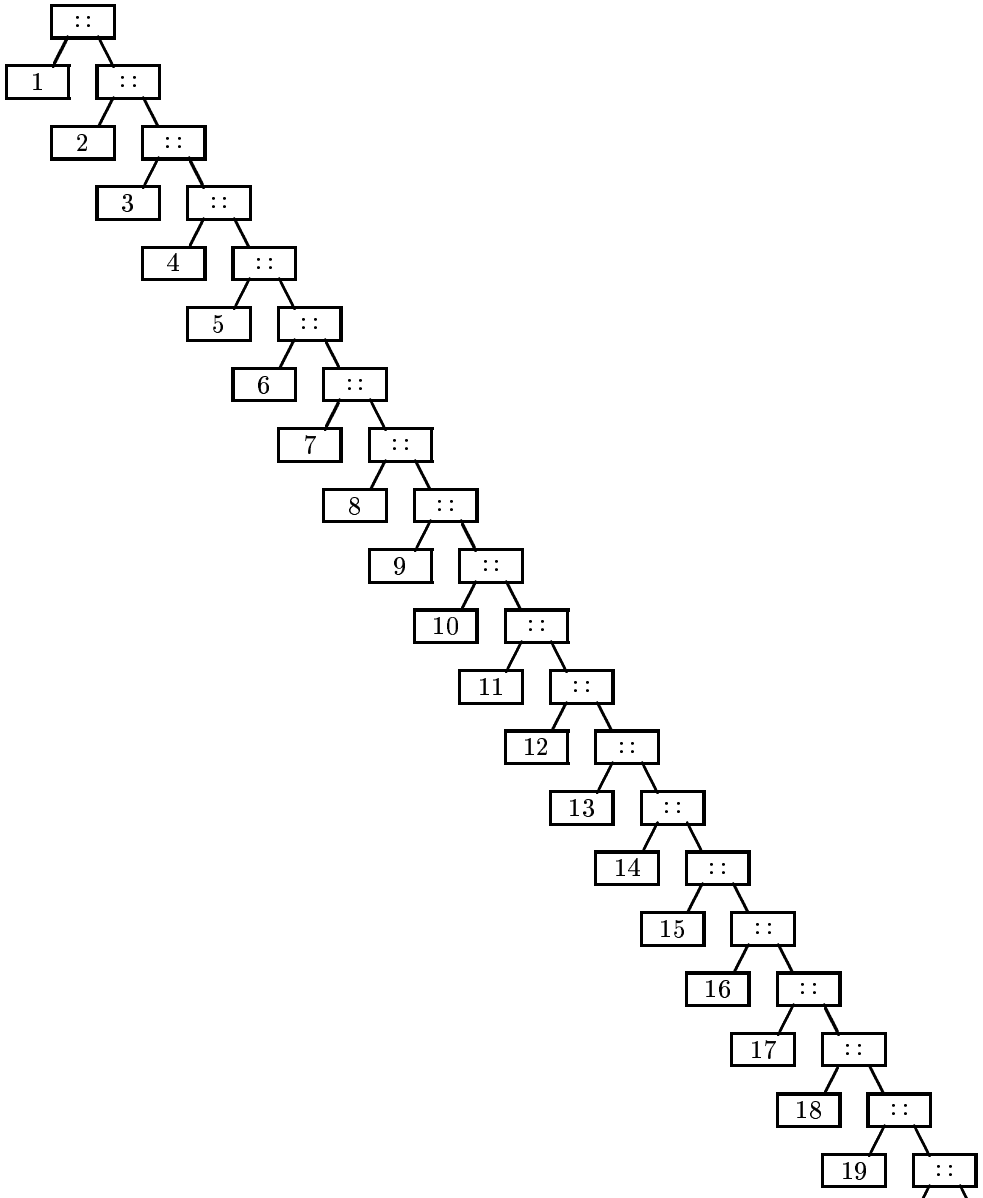
Answer 5.11.1

(a) See below.

(b) $[\,2\,]^{\cdot}$.

(c) $[\,\infty\text{-list}(1) \equiv \bot\,]^{\cdot}$ fails. (If $[\,\infty\text{-list}(1) \equiv \bot\,]^{\cdot}$ did hold then $[\,\infty\text{-list}(1)\ \text{head} \equiv \bot\,]^{\cdot}$ would also hold, but $[\,\infty\text{-list}(1)\ \text{head}\,]^{\cdot}$ equals $[\,1\,]^{\cdot}$).

(d) The computation takes infinitely long time since Map has to traverse all of $[\,\infty\text{-list}(1)\,]^{\cdot}$ to find the minimal element. Therefore, the value of $[\,\text{minlist}(\infty\text{-list}(1))\,]^{\cdot}$ is $[\,\bot\,]^{\cdot}$.



Answer 5.12.1

(a) $[\mathsf{T}]^{\cdot}$.

(b) $[\mathsf{F}]^{\cdot}$.

(c) $[\mathsf{F}]^{\cdot}$.

(d) $[\mathsf{F}]^{\cdot}$.

(e) $[\bullet]^{\cdot}$.

(f) $[\bot]^{\cdot}$.

(g) $[\bot]^{\cdot}$.

Chapter 6

Algebra

6.1 Algebraic rules

If $[\![x]\!]^{\cdot}$ and $[\![y]\!]^{\cdot}$ are arbitrary terms, then Map will compute

$$[\![(x :: y)\ \mathrm{head}]\!]^{\cdot}$$

and

$$[\![x]\!]^{\cdot}$$

to the same value. This is so because Map computes $[\![(x :: y)\ \mathrm{head}]\!]^{\cdot}$ by first reducing it to $[\![x]\!]^{\cdot}$ using $\left[\!\!\left[(x :: y)\ \mathrm{head} \xrightarrow{+} x \right]\!\!\right]^{\cdot}$ and then computing $[\![x]\!]^{\cdot}$. Therefore you have

$$[\![(x :: y)\ \mathrm{head} \equiv x]\!]^{\cdot}$$

for all terms $[\![x]\!]^{\cdot}$ and $[\![y]\!]^{\cdot}$.

As a special case, computation of $[\![x]\!]^{\cdot}$ may never end, but in this case both $[\![(x :: y)\ \mathrm{head}]\!]^{\cdot}$ and $[\![x]\!]^{\cdot}$ equal $[\![\bot]\!]^{\cdot}$, so the equation holds even in this case.

I will call $[\![(x :: y)\ \mathrm{head} \equiv x]\!]^{\cdot}$ an *algebraic rule*. In general, an algebraic rule is an equation that holds. Here you have eight algebraic rules:

$$\begin{array}{lcr}
[\![\langle\rangle\ \mathrm{head} & \equiv & \langle\rangle]\!]^{\cdot} \\
[\![(x :: y)\ \mathrm{head} & \equiv & x]\!]^{\cdot} \\
[\![\langle\rangle\ \mathrm{tail} & \equiv & \langle\rangle]\!]^{\cdot} \\
[\![(x :: y)\ \mathrm{tail} & \equiv & y]\!]^{\cdot} \\
[\![\mathrm{if}(T, x, y) & \equiv & x]\!]^{\cdot} \\
[\![\mathrm{if}(F, x, y) & \equiv & y]\!]^{\cdot} \\
[\![\langle\rangle\ \mathrm{atom} & \equiv & T]\!]^{\cdot} \\
[\![(x :: y)\ \mathrm{atom} & \equiv & F]\!]^{\cdot}
\end{array}$$

algebraic rule

The algebraic rules $[\![(x :: y)\ \text{head} \equiv x]\!]$ and $[\![(x :: y)\ \text{tail} \equiv y]\!]$ correspond to the reduction rules $\left[\!\left[(x :: y)\ \text{head} \xrightarrow{\pm} x \right]\!\right]$ and $\left[\!\left[(x :: y)\ \text{tail} \xrightarrow{\pm} y \right]\!\right]$, respectively. In general, all of the rules above correspond to reduction rules. Here you have four algebraic rules that do not correspond to reduction rules:

$$\begin{array}{ll}
[\![\bot\ \text{head} & \equiv\ \bot]\!] \\
[\![\bot\ \text{tail} & \equiv\ \bot]\!] \\
[\![\text{if}(\bot, x, y) & \equiv\ \bot]\!] \\
[\![\bot\ \text{atom} & \equiv\ \bot]\!]
\end{array}$$

6.2 Replacement

From $[\![(x :: y)\ \text{head} \equiv x]\!]$ you have that you can replace $[\![(x :: y)\ \text{head}]\!]$ by $[\![x]\!]$ in any context. As you may recall, this is known as "substitution of equals (c.f. Section 1.5). Here you have an example:

$$[\![(2 :: 3)\ \text{head} + 4 \equiv 2 + 4]\!].$$

In the example, I have replaced $[\![x]\!]$ by $[\![2]\!]$ and $[\![y]\!]$ by $[\![3]\!]$ in $[\![(x :: y)\ \text{head} \equiv x]\!]$ to get $[\![(2 :: 3)\ \text{head} \equiv 2]\!]$. Then I have replaced $[\![(2 :: 3)\ \text{head}]\!]$ by $[\![2]\!]$ in the context $[\![* + 4]\!]$.

I will say that

$$[\![(2 :: 3)\ \text{head} + 4 \equiv 2 + 4]\!]$$

is a *replacement* of $[\![(x :: y)\ \text{head} \equiv x]\!]$. Here you have five more examples of replacements of $[\![(x :: y)\ \text{head} \equiv x]\!]$:

$$\begin{array}{lcr}
[\![(x :: 4)\ \text{head} \cdot y & \equiv & x \cdot y]\!] \\
[\![x - (2 \cdot y :: z)\ \text{head} & \equiv & x - 2 \cdot y]\!] \\
[\![(y :: x)\ \text{head} & \equiv & y]\!] \\
[\![(x :: y)\ \text{head} & \equiv & x]\!] \\
[\![(2 :: 3)\ \text{head} + (2 :: 3)\ \text{head} & \equiv & (2 :: 3)\ \text{head} + 2]\!] \\
[\![(2 :: 3)\ \text{head} + (2 :: 3)\ \text{head} & \equiv & 2 + (2 :: 3)\ \text{head}]\!]
\end{array}$$

The following are not replacements of $[\![(x :: y)\ \text{head} \equiv x]\!]$:

$$\begin{array}{lcr}
[\![(2 :: 3)\ \text{head} & \equiv & 4]\!] \\
[\![(2 :: 3)\ \text{head} + 2 & \equiv & 2 \cdot 2]\!]
\end{array}$$

The first is not a replacement because the two $[\![x]\!]$'es in $[\![(x + y)\ \text{head} \equiv x]\!]$ are replaced by different values. The second is not a replacement because the context $[\![* + 2]\!]$ changes to $[\![* \cdot 2]\!]$. The second statement is an algebraic rule because it is an equation that holds (both sides of $[\![\equiv]\!]^\circ$ equal $[\![4]\!]$), but it is not a replacement of $[\![(x :: y)\ \text{head} \equiv x]\!]$.

replacement

The following are replacements of $[\,(x+y)\ \text{head} \equiv x\,]^{\cdot}$:

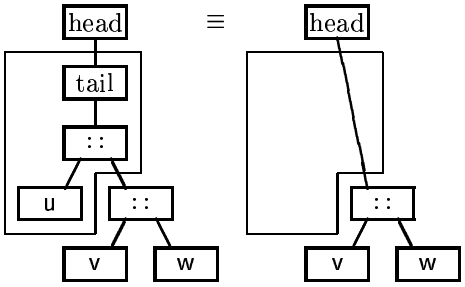
$$[\,(2::3)\ \text{head} + (4::5)\ \text{head} \equiv 2+4\,]^{\cdot}$$
$$[\,(2::3)\ \text{head} + (4::5)\ \text{head} \equiv (2::3)\ \text{head} + (4::5)\ \text{head}\,]^{\cdot}$$

In the first equation I make two replacements of $[\,(x::y)\ \text{head} \equiv x\,]^{\cdot}$. In the second equation I make zero replacements.

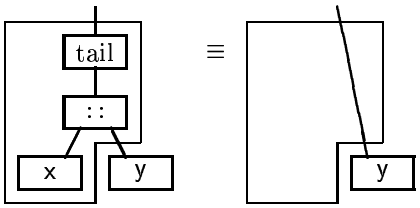
As a more complicated example,

$$[\,(u::v::w)\ \text{tail head} \equiv (v::w)\ \text{head}\,]^{\cdot}$$

is a replacement of $[\,(x::y)\ \text{tail} \equiv y\,]^{\cdot}$. The example is more complicated because of the parentheses and because the example depends on the right associativity of $[\,x::y\,]^{\cdot}$. The example becomes much simpler when written using syntax trees:



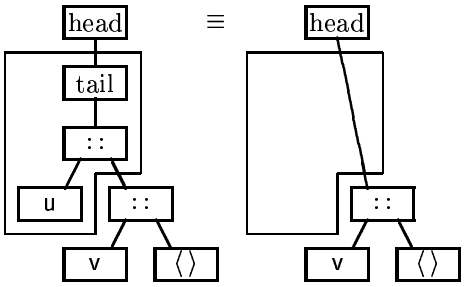
is a replacement of



As an even more complicated example,

$$[\,\langle u, v\rangle\ \text{tail head} \equiv \langle v\rangle\ \text{head}\,]^{\cdot}$$

is a replacement of $[\,(x::y)\ \text{tail} \equiv y\,]^{\cdot}$. Again, the example becomes simple using syntax trees:



The following are replacements of $[\![\, 0 \cdot \mathsf{x} \equiv 0 \,]\!]$:

$[\![\, (0 \cdot 1) \cdot (0 \cdot 2) + 117 \equiv 0 \cdot 0 + 117 \,]\!]$, and

$[\![\, 0 \cdot 0 + 117 \equiv 0 + 117 \,]\!]$.

The following is not a replacement of $[\![\, 0 \cdot \mathsf{x} \equiv 0 \,]\!]$:

$[\![\, (0 \cdot 1) \cdot (0 \cdot 2) + 117 \equiv 0 + 117 \,]\!]$.

One could argue that it is possible to get from $[\![\, (0 \cdot 1) \cdot (0 \cdot 2) + 117 \,]\!]$ to $[\![\, 0 + 117 \,]\!]$ by three replacements of $[\![\, 0 \cdot \mathsf{x} \,]\!]$ by $[\![\, 0 \,]\!]$, namely: replace $[\![\, 0 \cdot 1 \,]\!]$ by $[\![\, 0 \,]\!]$, replace $[\![\, 0 \cdot 2 \,]\!]$ by $[\![\, 0 \,]\!]$, then replace $[\![\, 0 \cdot 0 \,]\!]$ by $[\![\, 0 \,]\!]$. However, in multiple replacements I only allow replacements of redexes that occur in the original term, and $[\![\, 0 \cdot 0 \,]\!]$ does not occur in $[\![\, (0 \cdot 1) \cdot (0 \cdot 2) + 117 \,]\!]$.

Exercise 6.2.1 Which of the following are replacements of $[\![\, (\mathsf{x} :: \mathsf{y})\ \mathrm{tail} \equiv \mathsf{y} \,]\!]$:

(a) $[\![\, (3 :: 2)\ \mathrm{tail} - 4 \equiv 2 - 4 \,]\!]$.

(b) $[\![\, (\bot :: 2)\ \mathrm{tail} - \bot \equiv 2 - \bot \,]\!]$.

(c) $[\![\, \langle \mathsf{u}, \mathsf{v}, \mathsf{w} \rangle\ \mathrm{tail} = \mathsf{x} \equiv \langle \mathsf{v}, \mathsf{w} \rangle = \mathsf{x} \,]\!]$.

(d) $[\![\, (2 :: 3 :: 4)\ \mathrm{head}\ \mathrm{head} \equiv (2 :: 3)\ \mathrm{head} \,]\!]$.

(e) $[\![\, (2 :: 3 :: 4)\ \mathrm{head}\ \mathrm{head} \equiv (2 :: 3)\ \mathrm{tail} \,]\!]$.

(f) $[\![\, (2 :: 3 :: 4)\ \mathrm{head}\ \mathrm{tail} \equiv (2 :: 3)\ \mathrm{head} \,]\!]$.

(g) $[\![\, (2 :: 3 :: 4)\ \mathrm{head}\ \mathrm{tail} \equiv (2 :: 3)\ \mathrm{tail} \,]\!]$.

(h) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{head} \equiv (2 :: 3)\ \mathrm{head} \,]\!]$.

(i) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{head} \equiv (2 :: 3)\ \mathrm{tail} \,]\!]$.

(j) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{tail} \equiv (2 :: 3)\ \mathrm{head} \,]\!]$.

(k) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{tail} \equiv (2 :: 3)\ \mathrm{tail} \,]\!]$.

(l) $[\![\, (2 :: 3 :: 4)\ \mathrm{head}\ \mathrm{head} \equiv (3 :: 4)\ \mathrm{head} \,]\!]$.

(m) $[\![\, (2 :: 3 :: 4)\ \mathrm{head}\ \mathrm{head} \equiv (3 :: 4)\ \mathrm{tail} \,]\!]$.

(n) $[\![\, (2 :: 3 :: 4)\ \mathrm{head}\ \mathrm{tail} \equiv (3 :: 4)\ \mathrm{head} \,]\!]$.

(o) $[\![\, (2 :: 3 :: 4)\ \mathrm{head}\ \mathrm{tail} \equiv (3 :: 4)\ \mathrm{tail} \,]\!]$.

(p) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{head} \equiv (3 :: 4)\ \mathrm{head} \,]\!]$.

(q) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{head} \equiv (3 :: 4)\ \mathrm{tail} \,]\!]$.

(r) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{tail} \equiv (3 :: 4)\ \mathrm{head} \,]\!]$.

(s) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{tail} \equiv (3 :: 4)\ \mathrm{tail} \,]\!]$.

(t) $[\![\, (2 :: 3 :: 4)\ \mathrm{tail}\ \mathrm{tail} \equiv 4 \,]\!]$.

6.3 Reverse replacement

Here you have a *reverse replacement* of $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{head} \equiv \mathsf{x}\ \rceil$:

$$\lceil\, 2 + 4 \equiv (2 :: 3)\ \mathrm{head} + 4\ \rceil.$$

$\lceil\, 2 + 4 \equiv (2 :: 3)\ \mathrm{head} + 4\ \rceil$ holds according to substitution of equals, but I will refer to it as a reverse replacement rather than a replacement. In a replacement of $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{head} \equiv \mathsf{x}\ \rceil$, $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{head}\ \rceil$ is replaced by $\lceil\, \mathsf{x}\ \rceil$. In a reverse replacement, $\lceil\, \mathsf{x}\ \rceil$ is replaced by $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{head}\ \rceil$.

Exercise 6.3.1 Which of the following are reverse replacements of $\lceil\, \mathrm{if}(\mathsf{F}, \mathsf{x}, \mathsf{y}) \equiv \mathsf{y}\ \rceil$?

(a) $\lceil\, 2 + 3 \equiv \mathrm{if}(\mathsf{F}, 2, 4) + 3\ \rceil$.

(b) $\lceil\, 2 \cdot \mathsf{x} \equiv 2 \cdot \mathrm{if}(\mathsf{F}, \mathsf{y}, \mathsf{x})\ \rceil$.

(c) $\lceil\, 2 + \mathrm{if}(\mathsf{F}, 2, 3) \equiv 2 + 3\ \rceil$.

(d) $\lceil\, \mathsf{y} \equiv \mathrm{if}(\mathsf{F}, \mathsf{x}, \mathsf{y})\ \rceil$.

(e) $\left\lceil\, \mathsf{y} \equiv \mathsf{F}\left\{ \begin{array}{c} \mathsf{x} \\ \mathsf{y} \end{array} \right.\ \right\rceil$.

6.4 Algebraic proofs

From $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{head} \equiv \mathsf{x}\ \rceil$ and $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{tail} \equiv \mathsf{y}\ \rceil$ you can prove

$$\lceil\, (\mathsf{u} :: \mathsf{v} :: \mathsf{w})\ \mathrm{tail}\ \mathrm{head} :: \mathsf{y} \equiv \mathsf{v} :: (\mathsf{x} :: \mathsf{y})\ \mathrm{tail}\ \rceil$$

thus:

$$\begin{array}{lcl} \lceil\, (\mathsf{u} :: \mathsf{v} :: \mathsf{w})\ \mathrm{tail}\ \mathrm{head} :: \mathsf{y} & \equiv & (\mathsf{v} :: \mathsf{w})\ \mathrm{head} :: \mathsf{y} \\ & \equiv & \mathsf{v} :: \mathsf{y} \\ & \equiv & \mathsf{v} :: (\mathsf{x} :: \mathsf{y})\ \mathrm{tail}\ \rceil \end{array}$$

I will call this an *algebraic proof*. The proof consists of four terms separated by three equal signs. The proof proves that the first of the terms equals the last.

The proof consists of three *proof steps* (three equal signs). In the first step I replace $\lceil\, (\mathsf{u} :: \mathsf{v} :: \mathsf{w})\ \mathrm{tail}\ \rceil$ by $\lceil\, \mathsf{v} :: \mathsf{w}\ \rceil$ which is legal according to $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{tail} \equiv \mathsf{y}\ \rceil$. In the second step I replace $\lceil\, (\mathsf{v} :: \mathsf{w})\ \mathrm{head}\ \rceil$ by $\lceil\, \mathsf{v}\ \rceil$ which is legal according to $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{head} \equiv \mathsf{x}\ \rceil$. In the last step I replace $\lceil\, \mathsf{y}\ \rceil$ by $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{tail}\ \rceil$ which is legal according to $\lceil\, (\mathsf{x} :: \mathsf{y})\ \mathrm{tail} \equiv \mathsf{y}\ \rceil$.

---

reverse replacement
algebraic proof
proof step

A *lemma* is an equation that has a proof. Hence, $[\,(u :: v :: w)\ \text{tail head} :: y \equiv v :: (x :: y)\ \text{tail}\,]^{\cdot}$ is a lemma.

The phrases *theorem* and *main theorem* are synonyms for "lemma". It is common practice to use "theorem" to mean "particularly important lemma" and "main theorem" to mean "very important and very central lemma".

The phrase *conjecture* means "statement that somebody thinks is true, but for which no proof has been written yet". A conjecture turns into a lemma the moment somebody proves it.

6.5 Algebraic systems

Athens was a sovereign state in ancient times. It had a direct democracy and most citizens could read. The Athenians found it important for the democracy that all citizens had access to and could understand the law. For that reason, the law was written on barrels and placed on poles on the market place. The Athenians called poles something like *axioms*. A minority of mathematicians today find it important that mathematicians have access to and understand the foundations of mathematics. Such mathematicians do not place the fundamental laws of mathematics on poles, but they call the fundamental laws "axioms" to compensate.

The fundamental laws of mathematics are not fixed. Each mathematician is free to proclaim his or her set of laws. The literature calls such a set of laws an *axiomatic system*.

Some mathematicians are uninterested in foundations. Such mathematicians don't proclaim their own axiomatic systems. Rather, they use some well-established axiomatic system (maybe without even knowing the details of the system).

Other mathematicians have axiomatic systems as their topic of work. Such mathematicians may proclaim dozens of different axiomatic systems and they compare them and study their properties.

There are many brands of axiomatic systems. I will now present a kind of axiomatic system that I will refer to as *algebraic systems*. An algebraic system consists of a finite collection of constructs, a finite collection of algebraic rules, and a single *contradiction*. As an example, here is an algebraic system:

---

lemma
theorem
main theorem
conjecture
axiom
axiomatic system
algebraic system
contradiction

Constructs	Axioms			Contradiction
$[\,\langle\rangle\,]^{\cdot}$	$[\,\langle\rangle$ head	$\equiv$	$\langle\rangle\,]^{\cdot}$	$[\,\mathsf{T} \equiv \mathsf{F}\,]^{\cdot}$
$[\,\mathsf{x} :: \mathsf{y}\,]^{\cdot}$	$[\,(\mathsf{x} :: \mathsf{y})$ head	$\equiv$	$\mathsf{x}\,]^{\cdot}$	
$[\,\mathsf{x}$ head $]^{\cdot}$	$[\,\bot$ head	$\equiv$	$\bot\,]^{\cdot}$	
$[\,\mathsf{x}$ tail $]^{\cdot}$	$[\,\langle\rangle$ tail	$\equiv$	$\langle\rangle\,]^{\cdot}$	
$[\,\mathsf{x}$ atom $]^{\cdot}$	$[\,(\mathsf{x} :: \mathsf{y})$ tail	$\equiv$	$\mathsf{y}\,]^{\cdot}$	
$[\,\mathsf{T}\,]^{\cdot}$	$[\,\bot$ tail	$\equiv$	$\bot\,]^{\cdot}$	
$[\,\mathsf{F}\,]^{\cdot}$	$[\,\mathrm{if}(\mathsf{T},\mathsf{x},\mathsf{y})$	$\equiv$	$\mathsf{x}\,]^{\cdot}$	
$[\,\mathrm{if}(\mathsf{x},\mathsf{y},\mathsf{z})\,]^{\cdot}$	$[\,\mathrm{if}(\mathsf{F},\mathsf{x},\mathsf{y})$	$\equiv$	$\mathsf{y}\,]^{\cdot}$	
$[\,\bot\,]^{\cdot}$	$[\,\mathrm{if}(\bot,\mathsf{x},\mathsf{y})$	$\equiv$	$\bot\,]^{\cdot}$	
	$[\,\langle\rangle$ atom	$\equiv$	$\mathsf{T}\,]^{\cdot}$	
	$[\,(\mathsf{x} :: \mathsf{y})$ atom	$\equiv$	$\mathsf{F}\,]^{\cdot}$	
	$[\,\bot$ atom	$\equiv$	$\bot\,]^{\cdot}$	

Strictly speaking, the terms of the system are all terms like $[\,(\langle\rangle :: \bot)$ head head $]^{\cdot}$ that can be built up from the constructs. I will be a bit more liberal in the following, however, in that I will also allow terms to include variables like $[\,\mathsf{x}\,]^{\cdot}$ and $[\,\mathsf{y}\,]^{\cdot}$, numerals like $[\,1.23\,]^{\cdot}$ and $[\,117\mathsf{F}\,]^{\cdot}$, and defined concepts like $[\,\mathsf{x}+\mathsf{y}\,]^{\cdot}$ and $[\,\mathsf{x}\cdot\mathsf{y}\,]^{\cdot}$. With these relaxations, a term like $[\,\mathsf{x}+1 :: 2\,]^{\cdot}$ is also a term of the axiomatic system above.

Strictly speaking, the proofs of the system are sequences $[\,\mathcal{T}_0 \equiv \mathcal{T}_1 \equiv \cdots \equiv \mathcal{T}_\mathsf{n}\,]^{\circ}$ where $[\,\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_\mathsf{n}\,]^{\circ}$ are terms of the system and where each of the equations $[\,\mathcal{T}_0 \equiv \mathcal{T}_1, \mathcal{T}_1 \equiv \mathcal{T}_2, \ldots, \mathcal{T}_{\mathsf{n}-1} \equiv \mathcal{T}_\mathsf{n}\,]^{\circ}$ is either a replacement or a reverse replacement of an axiom of the system.

Strictly speaking, the lemmas of the system are equations of form $[\,\mathcal{R} \equiv \mathcal{S}\,]^{\cdot}$ where $[\,\mathcal{R}\,]^{\cdot}$ and $[\,\mathcal{S}\,]^{\cdot}$ are terms of the system and for which there exists a proof $[\,\mathcal{T}_0 \equiv \mathcal{T}_1 \equiv \cdots \equiv \mathcal{T}_\mathsf{n}\,]^{\circ}$ such that $[\,\mathcal{R}\,]^{\cdot}$ is the same term as $[\,\mathcal{T}_0\,]^{\cdot}$ and such that $[\,\mathcal{S}\,]^{\cdot}$ is the same term as $[\,\mathcal{T}_\mathsf{n}\,]^{\cdot}$ (two terms are the same if they have the same syntax tree).

An algebraic system is *inconsistent* (contradicts itself) if the contradiction is provable within the system. Hence, the system above is inconsistent if $[\,\mathsf{T} \equiv \mathsf{F}\,]^{\cdot}$ is provable. A system is *consistent* if it is not inconsistent. The system above is consistent since there is no way to prove $[\,\mathsf{T} \equiv \mathsf{F}\,]^{\cdot}$ within the system.

It is easier to prove a system inconsistent than consistent. To prove a system inconsistent you merely have to give a proof of the contradiction within the system itself. To prove a system consistent, you have to consider all proofs of the system and prove that none of them proves the contradiction. To prove the consistency of a system $[\,\mathcal{P}\,]^{\cdot}$, you need a system $[\,\mathcal{Q}\,]^{\cdot}$ in which you can talk about "all proofs of $[\,\mathcal{P}\,]^{\cdot}$". In 1931, Kurt Gödel published a result which you can interpret as follows: If the consistency of $[\,\mathcal{P}\,]^{\cdot}$ is provable in $[\,\mathcal{Q}\,]^{\cdot}$, then $[\,\mathcal{Q}\,]^{\cdot}$ is strictly more powerful than $[\,\mathcal{P}\,]^{\cdot}$. In particular, no system can prove its own consistency. The exact formulation of Gödel's result is a bit involved. Formulations are crucial here, but I will not go into details.

---

inconsistent
consistent

It is possible to construct an algebraic system that is so small that it can be written on a single page and so powerful that all lemmas of classical mathematics can be proved within it. This is true even for the strictest interpretation of the notion of "algebraic system". When "algebraic system" is interpreted strictly, even the notion of a variable has to be introduced by finitely many constructs and described by finitely many axioms. Such a minimalistic algebraic system is nice to have in certain situations. Minimalistic algebraic systems are particularly useful for mathematicians who have axiomatic systems as their topic of work. Until further, I will not try to be minimalistic.

6.6 [Mac]· rules

You are free to proclaim your own algebraic system, but in this book I proclaim my system, and I expect you to use my system as long as you read my book. I will refer to my algebraic system as the [Mac]· system. [Mac]· stands for "mathematics and computation", which happens to be the title of this book.

I am going to tell both you and Map about the [Mac]· system. I tell Map about the [Mac]· system because that allows Map to proofread my algebraic proofs. First, I tell Map that there is something called [Mac]·:

[**construct** Mac].

In Section 6.1 I explained why I consider [(x :: y) head ≡ x]· a valid rule. I now tell Map that [(x :: y) head ≡ x]· is an axiom of the [Mac]· system:

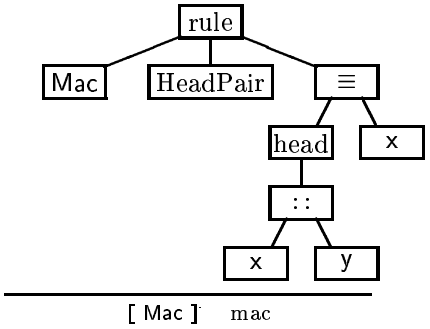
[**Mac rule HeadPair** : (x :: y) head ≡ x]

The statement above has two effects. First, the statement tells Map that [(x :: y) head ≡ x]· is an axiom within the [Mac]· system. Second, the statement introduces [HeadPair]· as a new construct. The statement defines [HeadPair]· as a macro that reduces to [(x :: y) head ≡ x]·.

The syntax tree of

[**Mac rule HeadPair** : (x :: y) head ≡ x]

reads:



I will say that [(x :: y) head ≡ x]⁻ is a [Mac]⁻ axiom.

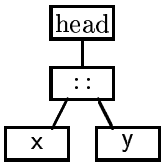
I will also say that [(x :: y) head ≡ x]⁻ is a [Mac]⁻ *rule*. Until further (until Section 11.2), you can take "axiom" and "rule" as synonyms.

In general, I proclaim rules using the construct

[x **rule** y:z]⁻

where [z]⁻ is the rule I proclaim, [y]⁻ is the name of the rule, and [x]⁻ is the axiomatic system the rule belongs to.

The syntax tree of the left hand side of [HeadPair]⁻ reads:



As you can see, the root of the left hand side of the axiom reads "head". In other words, [x head]⁻ is the principal operator of the left hand side of the axiom. If you look up [HeadPair]⁻ in Appendix A, you will find [HeadPair]⁻ in the section that deals with [x head]⁻. In general, I try to place mac rules those places where I expect you to look for them. If you look for a particular mac rule and if you don't know exactly where in Appendix A I have placed it, you must try to guess where I expect you to look. The principal operator of the left hand side of the conclusion is one of the places I like to put mac rules. In general, I try to place [Mac]⁻ rules under the "most characteristic" operator of the axiom, and that operator is often (but not always) the principal operator.

If you know the name of a [Mac]⁻ axiom, you can use the index to find the definition of the axiom.

6.7 [Mac]⁻ lemmas

In Section 6.4 I proved the lemma

[(u :: v :: w) tail head :: y ≡ v :: (x :: y) tail]⁻.

I now state and prove the lemma above more formally. First, I state the lemma formally:

[**Mac lemma L6.7.1** : (u :: v :: w) tail head :: y ≡ v :: (x :: y) tail]

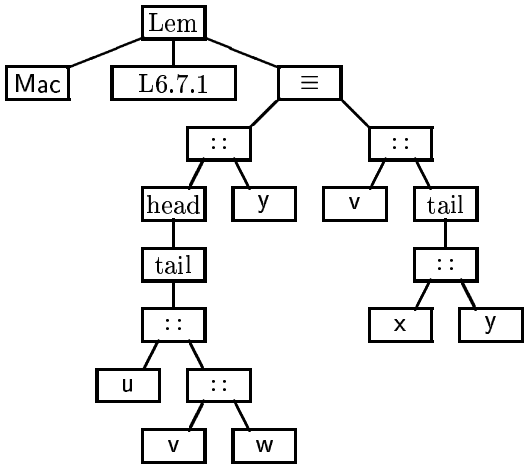
The statement above has two effects. First, the statement says that somewhere in this book there is a proof of [(u :: v :: w) tail head :: y ≡ v :: (x :: y) tail]⁻. Second, the statement introduces [L6.7.1]⁻ as a macro that reduces to [(u :: v :: w) tail head :: y ≡ v :: (x :: y) tail]⁻.

The syntax tree of

	rule
[x **rule** y:z]⁻	x rule y colon z

[**Mac lemma L6.7.1** : (u :: v :: w) tail head :: y ≡ v :: (x :: y) tail]

reads:



In general, I proclaim lemmas using the construct

[[x **lemma** y:z]].

where [z]˙ is the lemma I proclaim, [y]˙ is the name of the lemma, and [x]˙ is the axiomatic system in which the lemma is going to be proved.

6.8 [Mac]˙ proofs

I Section 6.7 I promissed to give a proof of [(u :: v :: w) tail head :: y ≡ v :: (x :: y) tail]˙. I formulated the promise thus:

[**Mac lemma L6.7.1** : (u :: v :: w) tail head :: y ≡ v :: (x :: y) tail]

Map allows me to repeat the promise any number of times, but Map would complain if I didn't keep my promise. I keep my promise thus:

[**Mac proof of** (u :: v :: w) tail head :: y ≡ v :: (x :: y) tail**:**

Algebra ▷	(u :: v :: w) tail head :: y	;
Replace ▷ (x :: y) tail ≡ y ▷	(v :: w) head :: y	;
Replace ▷ (x :: y) head ≡ x ▷	v :: y	;
Reverse ▷ (x :: y) tail ≡ y ▷	v :: (x :: y) tail	]

In this book you will meet two kinds of proofs: algebraic proofs and derivations. You will meet derivations in Chapter 11. Here in Chapter 6 I will merelly

[x **lemma** y:z]˙ x lemma y colon z

consider algebraic proofs. You can see that the proof above is algebraic because of the word $\lceil\boxed{\text{Algebra}}\rceil$ in the second line.

In Section 6.4 I gave an informal proof of $\lceil$ (u :: v :: w) tail head :: y ≡ v :: (x :: y) tail $\rceil$:

$$\begin{array}{rcl} \lceil\ (\mathsf{u} :: \mathsf{v} :: \mathsf{w})\ \mathsf{tail}\ \mathsf{head} :: \mathsf{y} & \equiv & (\mathsf{v} :: \mathsf{w})\ \mathsf{head} :: \mathsf{y} \\ & \equiv & \mathsf{v} :: \mathsf{y} \\ & \equiv & \mathsf{v} :: (\mathsf{x} :: \mathsf{y})\ \mathsf{tail}\ \rceil \end{array}$$

The informal proof consists of four terms separated by three equal signs. The formal proof above consists of the same four terms plus some formalities.

The overall structure of the proof reads:

$$\lceil\boxed{\mathcal{S}\ \textbf{proof of}\ \mathcal{L}\textbf{:}\mathcal{B}}\rceil$$

where $\lceil\ \mathcal{S}\ \rceil$ is the algebraic system I have chosen to use in the proof, $\lceil\ \mathcal{L}\ \rceil$ is the lemma to be proved, and $\lceil\ \mathcal{B}\ \rceil$ is the body of the proof. In the proof above, the algebraic system is $\lceil$ Mac $\rceil$, the lemma is $\lceil$ (u :: v :: w) tail head :: y ≡ v :: (x :: y) tail $\rceil$, and the body consists of four lines.

The structure of the body is $\lceil\ \mathcal{D};\mathcal{E};\mathcal{F};\mathcal{G}\ \rceil$ where I refer to $\lceil\ \mathcal{D}\ \rceil$, $\lceil\ \mathcal{E}\ \rceil$, $\lceil\ \mathcal{F}\ \rceil$, and $\lceil\ \mathcal{G}\ \rceil$ as *proof lines*. As you can see, I use the construct $\lceil\boxed{\mathsf{x;y}}\rceil$ to glue proof lines together.

Each proof line has the structure

$$\lceil\ \mathcal{A}\boxed{\triangleright}\mathcal{T}\ \rceil$$

where $\lceil\ \mathcal{A}\ \rceil$ is an *argumentation* and $\lceil\ \mathcal{T}\ \rceil$ is a term. In algebraic proofs, the argumentation of the first proof line reads $\lceil$ Algebra $\rceil$. In pure algebraic proofs, the argumentations of the remaining proof lines have form either

$$\lceil\boxed{\text{Replace}}\triangleright\mathcal{X}\rceil \text{ or}$$

$$\lceil\boxed{\text{Reverse}}\triangleright\mathcal{X}\rceil$$

where $\lceil\ \mathcal{X}\ \rceil$ must be an axiom within the chosen algebraic system. For convenience, I will introduce further argumentations later on, but it is nice to know that all of mathematics can be developed on basis of $\lceil$ Replace $\rceil$ and $\lceil$ Reverse $\rceil$ alone.

$\lceil$ Algebra $\rceil$	Algebra
$\lceil$ x **proof of** y**:**z $\rceil$	x proof of y colon z
	proof line
$\lceil$ x;y $\rceil$	x semicolon y
$\lceil$ x ▷ y $\rceil$	x concludes y
	argumentation
$\lceil$ Replace $\rceil$	replace
$\lceil$ Reverse $\rceil$	reverse

In the formal proof above, the first and second proof line together say that

$$[\,(u :: v :: w)\ \text{tail head} :: y \equiv (v :: w)\ \text{head} :: y\,]$$

is a replacement of

$$[\,(x :: y)\ \text{tail} :: y\,].$$

Note that $[\,(u :: v :: w)\ \text{tail head} :: y\,]$ is the term of the first proof line, $[\,(x :: y)\ \text{tail} \equiv y\,]$ is the argumentation of the third line, and $[\,(v :: w)\ \text{head} :: y\,]$ is the term of the second proof line.

The second and third proof line say that

$$[\,(v :: w)\ \text{head} :: y \equiv v :: y\,]$$

is a replacement of

$$[\,(x :: y)\ \text{head} \equiv x\,].$$

The third and fourth proof line say that

$$[\,v :: y \equiv v :: (x :: y)\ \text{tail}\,]$$

is a replacement of

$$[\,(x :: y)\ \text{tail} \equiv y\,].$$

6.9 Use of abbreviations

I will prove

$$[\,\textbf{Mac lemma L6.7.1}\ :\ (u :: v :: w)\ \text{tail head} :: y \equiv v :: (x :: y)\ \text{tail}\,]$$

once more:

[Mac proof of L6.7.1**:**

Algebra ▷	$(u :: v :: w)$ tail head $:: y$	;
Replace ▷ TailPair ▷	$(v :: w)$ head $:: y$	;
Replace ▷ HeadPair ▷	$v :: y$	;
Reverse ▷ TailPair ▷	$v :: (x :: y)$ tail	]

In the proof above I have used that [L6.7.1] is shorthand for $[\,(u :: v :: w)\ \text{tail head} :: y \equiv v :: (x :: y)\ \text{tail} \equiv v :: (x :: y)\ \text{tail}\,]$, that [HeadPair] is shorthand for $[\,(x :: y)\ \text{head} \equiv y\,]$, and that [TailPair] is shorthand for $[\,(x :: y)\ \text{tail} \equiv y\,]$.

You can look up [HeadPair] and [TailPair] in the index. The index will direct you to the pages of the reference manual where I proclaim [HeadPair] and [TailPair].

6.10 The [Mac]· contradiction

In Section 6.5 I stated that an algebraic system consists of a collection of constructs, a collection of axioms, and a contradiction. I state the axioms of the [Mac]· system in the reference manual using statements of form

[Mac rule y:z]·.

Until further, I will not be specific about which constructs [Mac]· has, but I will state that the contradiction of the system reads [$\bot \equiv \mathsf{T}$]·:

[Mac antirule Contradiction : $\bot \equiv \mathsf{T}$]

The statement above tells Map that [$\bot \equiv \mathsf{T}$]· is the contradiction of the [Mac]· system and also introduces [Contradiction]· as shorthand for [$\bot \equiv \mathsf{T}$]·. In Chapter 16 I will show how the contradiction allows to disprove statements, i.e. to prove that given statements fail. Until further, I will concentrate on methods for proving statements.

6.11 Algebra versus computation

Algebraic proofs are somewhat similar to reductions. As an example,

$$[\ (1 :: 2 :: 3)\ \text{tail head} \xrightarrow{+} (2 :: 3)\ \text{head} \xrightarrow{+} 2\ ]^{\cdot}$$

is a reduction and

$$[\ (1 :: 2 :: 3)\ \text{tail head} \equiv (2 :: 3)\ \text{head} \equiv 2\ ]^{\cdot}$$

is an algebraic proof. In general, you can change a reduction into an algebraic proof by replacing $[\ \xrightarrow{+}\ ]^{\circ}$ by $[\ \equiv\ ]^{\circ}$.

Reductions are much more restricted than algebraic proofs, however. If you supply the first term of a reduction, then a computer can produce the rest of the reduction. Hence, reductions are *deterministic* in that each term can be derived from the previous in a purely mechanical manner. Furthermore, reduction rules can only be used from left to right.

Algebraic proofs are *non-deterministic*. If the first term of a proof is [(1 :: 2 :: 3) tail head]·, then the next could be [(2 :: 3) head]· as above, but it could also be [(0 :: 1 :: 2 :: 3) tail tail head]·, depending on the mood of the mathematician who wrote it.

A computer can check the correctness of algebraic proofs very efficiently. It simply has to run through the proof and check that each step is an instance of a rule of the algebraic system in use.

In principle, one can also ask a computer to search for a proof of equality of two terms. Computers, at least at present, however, are very bad at searching for proofs; they are typically unable to find proofs even for simple theorems and even if they get billions of years for the search.

deterministic
non-deterministic

6.12 Proofs and syntax trees

Here you have a lemma and a proof:

[**Mac lemma L6.12.1** : $(1 :: 2 :: \langle\rangle)$ tail head $\equiv 2$]

[**Mac proof of** L6.12.1**:**

Algebra $\triangleright$	$(1 :: 2 :: \langle\rangle)$ tail head	;
Replace $\triangleright$ $(x :: y)$ tail $\equiv y \triangleright$	$(2 :: \langle\rangle)$ head	;
Replace $\triangleright$ $(x :: y)$ head $\equiv x \triangleright$	2	]

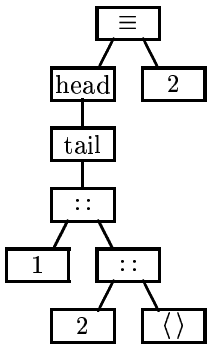
Here you have the same lemma and the same proof written another way:

[**Mac lemma L6.12.2** : $\langle 1, 2\rangle$ tail head $\equiv 2$]

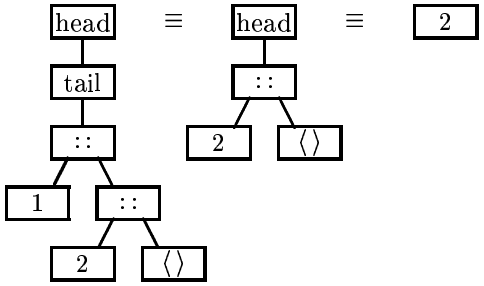
[**Mac proof of** L6.12.2**:**

Algebra $\triangleright$	$\langle 1, 2\rangle$ tail head	;
Replace $\triangleright$ $(x :: y)$ tail $\equiv y \triangleright$	$\langle 2\rangle$ head	;
Replace $\triangleright$ $(x :: y)$ head $\equiv x \triangleright$	2	]

When you read lemmas and proofs you should think of terms as shorthand for syntax trees. Both of the lemmas above state the following:



Both of the proofs state the following:



Map converts lemmas and proofs to syntax trees before checking them. In general, you must convert lemmas and proofs to syntax trees to see exactly what they say.

6.13 Reflexivity

Here you have a lemma with a somewhat peculiar proof:

[Mac lemma L6.13.1 : $2 \equiv 2$ **]**

[Mac proof of L6.13.1**:**
Algebra ▷ 2 **]**

The proof is a zero step proof. It consists of only one proof line. The conclusion of the proof is that the first term in the proof equals the last. The proof merely contains one term, namely $[\ 2\ ]$. Therefore, $[\ 2\ ]$ is both the first and the last term of the proof so the conclusion of the proof is $[\ 2 \equiv 2\ ]$.

In general you have $[\ \mathsf{x} \equiv \mathsf{x}\ ]$ for all terms $[\ \mathsf{x}\ ]$. The literature refers to $[\ \mathsf{x} \equiv \mathsf{x}\ ]$ as the *axiom of reflexivity*.

Here you have a more complicated example:

[Mac lemma L6.13.2 : $\langle 1, 2, 3\rangle \equiv 1 :: 2 :: 3 :: \langle\rangle$ **]**

[Mac proof of L6.13.2**:**
Algebra ▷ $\langle 1, 2, 3\rangle$ **]**

In other words, $[\ \langle 1, 2, 3\rangle \equiv 1 :: 2 :: 3 :: \langle\rangle\ ]$ holds by reflexivity because $[\ \langle 1, 2, 3\rangle\ ]$ and $[\ 1 :: 2 :: 3 :: \langle\rangle\ ]$ have identical syntax trees.

It is sometimes difficult to understand proofs simply because of notation. For that reason I extend the collection of argumentations with one I call

$$\Big[\ \boxed{\text{Reflexivity}}\ \Big].$$

Here you have a lemma with a proof that uses reflexivity:

[Mac lemma L6.13.3 : $\langle \mathsf{u}, \mathsf{v}, \mathsf{w}\rangle$ tail head $\equiv \mathsf{v}$ **]**

[Mac proof of L6.13.3**:**

Algebra ▷	$\langle \mathsf{u}, \mathsf{v}, \mathsf{w}\rangle$ tail head	;
Reflexivity ▷	$(\mathsf{u} :: \langle \mathsf{v}, \mathsf{w}\rangle)$ tail head	;
Replace ▷ $(\mathsf{x} :: \mathsf{y})$ tail $\equiv \mathsf{y}$ ▷	$\langle \mathsf{v}, \mathsf{w}\rangle$ head	;
Reflexivity ▷	$(\mathsf{v} :: \langle \mathsf{w}\rangle)$ head	;
Replace ▷ $(\mathsf{x} :: \mathsf{y})$ head $\equiv \mathsf{x}$ ▷	v	**]**

After term reduction, the first and second term of the proof are identical, so they are equal according to reflexivity. Hence, from the formal point of view, step $[\ 1\ ]$ in the proof above is valid because, after term reduction, the two sides of the equal sign are identical. A human reader, however, may see the first step as an instance of the term reduction rule

$$\frac{\big[\ \langle \mathsf{x}, \mathsf{y}\rangle \overset{\circ}{\to} \mathsf{x} :: \langle \mathsf{y}\rangle\ \big].}{\big[\ \text{Reflexivity}\ \big]}$$

axiom of reflexivity
reflexivity

A mechanical proof reader like Map will perform term reduction on a proof like the one above and will accept the first step because the two sides of the equal sign are identical, but a human reader will see that the first step follows immediately from the term reduction rule. It is possible to write proofs that are straightforward to read for mechanical proof readers and difficult for human proof readers. It is also possible to write proofs that are easy to verify for human readers but for which term reduction produces so large terms that mechanical proof readers have to give up due to lack of memory resources. I would say that a proof is good if both human and mechanical proofreaders find it easy to understand.

Above, you have seen how term reduction rules can enter proofs through reflexivity. You may also use reflexivity to express applications of macro definitions:

[Mac lemma L6.13.4 : $\langle \mathsf{x} \rangle$ tail $\equiv \langle\,\rangle$ **]**

[Mac proof of L6.13.4**:**

Algebra $\triangleright$	$\langle \mathsf{x} \rangle$ tail	;
Reflexivity $\triangleright$	$(\mathsf{x} :: \langle\,\rangle)$ tail	;
Replace $\triangleright$ $(\mathsf{x} :: \mathsf{y})$ tail $\equiv$ y $\triangleright$	$\langle\,\rangle$	**]**

The first step makes use of the macro definition

$$[\ \langle \mathsf{x} \rangle \doteq \mathsf{x} :: \langle\,\rangle\ ].$$

Exercise 6.13.5 Prove the following:

[Mac lemma L6.13.6 : $\langle 2, 3 \rangle$ tail tail $\equiv \langle\,\rangle$ **]**

6.14 Definitions

Each definition introduces, among other, a new axiom which I will refer to as *definition axioms*. As an example, the definition

$$\left[\ \boxed{\mathsf{x\ append\ y}} \doteq \mathsf{x\ atom}\left\{\begin{array}{l}\mathsf{y}\\ \mathsf{x\ head} :: (\mathsf{x\ tail\ append\ y})\end{array}\right.\ \right]$$

introduces the axiom

$$\left[\ \mathsf{x\ append\ y} \equiv \mathsf{x\ atom}\left\{\begin{array}{l}\mathsf{y}\\ \mathsf{x\ head} :: (\mathsf{x\ tail\ append\ y})\end{array}\right.\ \right]^{\cdot}.$$

For an algebraic system in the strictest sense, the collection of axioms is fixed. Hence, fundamentalists do not tolerate definitions. Until further I will use definitions and the axioms they introduce freely.

[x append y]$^{\cdot}$ definition axiom
x append y

Fundamentalists see no problem in macro definitions and term reduction rules as they do not introduce axioms and do not introduce new constructs.

I give $[\,\mathsf{x}\ \text{append}\ \mathsf{y}\,]^{\cdot}$ priority and associativity thus:

$[\,\mathsf{x} :: \mathsf{y} \overset{\smile}{>} \mathsf{x}\ \text{append}\ \mathsf{y} \overset{\smile}{>} \mathsf{x}, \mathsf{y}\,]$, and

$[\,\mathsf{x}\ \text{append}\ \mathsf{y}\ \text{append}\ \mathsf{z} \overset{\smile}{\to} \mathsf{x}\ \text{append}\ (\mathsf{y}\ \text{append}\ \mathsf{z})\,]$.

Here you have an example of use:

$[\,$**Mac lemma L6.14.1** $: \langle\rangle\ \text{append}\ \mathsf{y} \equiv \mathsf{y}\,]$

$[\,$**Mac proof of** L6.14.1**:**

Algebra $\triangleright$	$\langle\rangle\ \text{append}\ \mathsf{y}$	;
Replace $\triangleright$		
$\mathsf{x}\ \text{append}\ \mathsf{y} \equiv$		
$\text{if}(\mathsf{x}\ \text{atom}, \mathsf{y}, \mathsf{x}\ \text{head} ::$		
$(\mathsf{x}\ \text{tail append y})) \triangleright$	$\langle\rangle\ \text{atom}\left\{\begin{array}{l}\mathsf{y}\\ \langle\rangle\ \text{head} :: (\langle\rangle\ \text{tail append y})\end{array}\right.$	;
Replace $\triangleright \langle\rangle\ \text{atom} \equiv \mathsf{T} \triangleright$	$\mathsf{T}\left\{\begin{array}{l}\mathsf{y}\\ \langle\rangle\ \text{head} :: (\langle\rangle\ \text{tail append y})\end{array}\right.$	;
Replace $\triangleright\ \text{if}(\mathsf{T}, \mathsf{x}, \mathsf{y}) \equiv \mathsf{x} \triangleright$	y	$]$

The argumentation

$$\left[\,\text{Replace} \triangleright \mathsf{x}\ \text{append}\ \mathsf{y} \equiv \mathsf{x}\ \text{atom}\left\{\begin{array}{l}\mathsf{y}\\ \mathsf{x}\ \text{head} :: (\mathsf{x}\ \text{tail append y})\end{array}\right.\,\right]^{\cdot}$$

in the second proof line above is rather cumbersome to write down. For that reason I introduce a new argumentation that reads

$$[\,\boxed{\text{Definition}}\,]^{\cdot}.$$

$[\,\text{Definition}\,]^{\cdot}$ means either $[\,\text{Replace} \triangleright \mathcal{A} \equiv \mathcal{B}\,]^{\cdot}$ or $[\,\text{Reverse} \triangleright \mathcal{A} \equiv \mathcal{B}\,]^{\cdot}$ for some definition axiom of form $[\,\mathcal{A} \equiv \mathcal{B}\,]^{\cdot}$. Here you have three argumentations that can all be abbreviated into $[\,\text{Definition}\,]^{\cdot}$:

$$\left[\,\text{Replace} \triangleright \mathsf{x}\ \text{append}\ \mathsf{y} \equiv \mathsf{x}\ \text{atom}\left\{\begin{array}{l}\mathsf{y}\\ \mathsf{x}\ \text{head} :: (\mathsf{x}\ \text{tail append y})\end{array}\right.\,\right]^{\cdot},$$

$$\left[\,\text{Reverse} \triangleright \mathsf{x}\ \text{append}\ \mathsf{y} \equiv \mathsf{x}\ \text{atom}\left\{\begin{array}{l}\mathsf{y}\\ \mathsf{x}\ \text{head} :: (\mathsf{x}\ \text{tail append y})\end{array}\right.\,\right]^{\cdot}, \text{ and}$$

$[\,\text{Replace} \triangleright f_3(\mathsf{x}) \equiv 2 \cdot \mathsf{x} + 4)\,]^{\cdot}$.

Here you have the proof above once more:

$[\,\text{Definition}\,]^{\cdot}$	definition

[**Mac proof of** L6.14.1**:**

Algebra ▷	$\langle\rangle$ append y	;
Definition ▷	$\langle\rangle$ atom $\begin{cases} \mathsf{y} \\ \langle\rangle \text{ head} :: (\langle\rangle \text{ tail append y}) \end{cases}$	;
Replace ▷ $\langle\rangle$ atom ≡ T ▷	$\mathsf{T}\begin{cases} \mathsf{y} \\ \langle\rangle \text{ head} :: (\langle\rangle \text{ tail append y}) \end{cases}$	;
Replace ▷ if(T, x, y) ≡ x ▷	y	]

See Section A.18 for the rule [if(T, x, y) ≡ x]. See Section 6.16 for the rule [$\langle\rangle$ atom ≡ T].

Here you have one more example of use:

[**Mac lemma L6.14.2** : x :: y append z ≡ x :: (y append z)]

[**Mac proof of** L6.14.2**:**

Algebra ▷	x :: y append z	;
Definition ▷	if((x :: y) atom, z, (x :: y) head :: ((x :: y) tail append z))	;
Replace ▷ (x :: y) atom ≡ F ▷	if(F, z, (x :: y) head :: ((x :: y) tail append z))	;
Replace ▷ if(F, x, y) ≡ y ▷	(x :: y) head :: ((x :: y) tail append z)	;
Replace ▷ (x :: y) head ≡ x ▷	x :: ((x :: y) tail append z)	;
Replace ▷ (x :: y) tail ≡ y ▷	x :: (y append z)	]

Exercise 6.14.3 I define

$$\left[\ \mathsf{x}\ \text{two} \doteq \mathsf{x}\ \text{atom}\begin{cases} 2 \\ \mathsf{x}\ \text{tail two} \end{cases}\ \right].$$

You prove

[**Mac lemma L6.14.4** : $\langle\rangle$ two ≡ 2.]

6.15 Use of lemmas

Each lemma introduces, among other, a new axiom which I will refer to as *lemma axioms*. As an example, Lemma [L6.14.1] introduces the axiom

[$\langle\rangle$ append y ≡ y].

Again, fundamentalists will say that the collection of axioms is fixed and that lemmas do not introduce new axioms. Until further, I will be relaxed on this point and say that each time a conjecture is proved, the conjecture becomes a lemma and, thereby, a new axiom.

Here you have a lemma that uses Lemmas [L6.14.1] and [L6.14.2]:

lemma axiom

[Mac lemma L6.15.1 : $\langle x, y\rangle$ append $\langle u, v\rangle \equiv \langle x, y, u, v\rangle$ **]**

[Mac proof of L6.15.1**:**

Algebra ▷	$\langle x, y\rangle$ append $\langle u, v\rangle$	;
Reflexivity ▷	$(x :: \langle y\rangle)$ append $\langle u, v\rangle$	;
Replace ▷		
$x :: y$ append $z \equiv x :: (y$ append $z)$ ▷	$x :: (\langle y\rangle$ append $\langle u, v\rangle)$	;
Reflexivity ▷	$x :: (y :: \langle\,\rangle$ append $\langle u, v\rangle)$	;
Replace ▷		
$x :: y$ append $z \equiv x :: (y$ append $z)$ ▷	$x :: y :: (\langle\,\rangle$ append $\langle u, v\rangle)$	;
Replace ▷ $\langle\,\rangle$ append $y \equiv y$ ▷	$x :: y :: \langle u, v\rangle$	;
Reflexivity ▷	$\langle x, y, u, v\rangle$	**]**

Or, using shorthand notation:

[Mac proof of L6.15.1**:**

Algebra ▷	$\langle x, y\rangle$ append $\langle u, v\rangle$	;
Reflexivity ▷	$(x :: \langle y\rangle)$ append $\langle u, v\rangle$	;
Replace ▷ L6.14.2 ▷	$x :: (\langle y\rangle$ append $\langle u, v\rangle)$	;
Reflexivity ▷	$x :: (y :: \langle\,\rangle$ append $\langle u, v\rangle)$	;
Replace ▷ L6.14.2 ▷	$x :: y :: (\langle\,\rangle$ append $\langle u, v\rangle)$	;
Replace ▷ L6.14.1 ▷	$x :: y :: \langle u, v\rangle$	;
Reflexivity ▷	$\langle x, y, u, v\rangle$	**]**

As you can see, I use [L6.14.1]˙ and [L6.14.2]˙ as axioms in the arguments in proof line [3]˙, [5]˙, and [6]˙ above.

6.16 Computation

In addition to the extra axioms that definitions and lemmas provide, I will introduce an infinitude of *computation axioms*. As an example,

$$[\ 2 + 2 \equiv 4\ ]^{\cdot}$$

is an axiom of computation. It is an axiom of computation because if Map computes [2 + 2]˙ and [4]˙, then Map gets the same result in the two cases. The axiom of computation merely applies to equations that contain no variables. As an example, [$\langle x, y\rangle$ tail head $\equiv$ y]˙ is not an axiom of computation because it contains the variables [x]˙ and [y]˙.

I now give an example of use. First, I define

$$\left[\ x \text{ repeat } y \doteq y = 0 \left\{ \begin{array}{l} \langle\,\rangle \\ x :: (x \text{ repeat } y - 1) \end{array} \right.\right],$$

[x repeat y repeat z $\overset{\smile}{\to}$ (x repeat y) repeat z], and

computation axiom

$$[\ \mathsf{x} :: \mathsf{y} \mathbin{\breve{>}} \mathsf{x}\ \text{repeat}\ \mathsf{y} \mathbin{\breve{>}} \mathsf{x}\ \text{append}\ \mathsf{y}\ ].$$

Then, I state and prove the following:

[Mac lemma L6.16.1 : x repeat $2 \equiv \langle \mathsf{x}, \mathsf{x} \rangle$ **]**

[Mac proof of L6.16.1**:**

Algebra $\triangleright$	x repeat 2	;
Definition $\triangleright$	$2 = 0 \left\{ \begin{array}{l} \langle\,\rangle \\ \mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 2 - 1) \end{array} \right.$	;
Replace $\triangleright\ 2 = 0 \equiv \mathsf{F}\ \triangleright$	$\mathsf{F} \left\{ \begin{array}{l} \langle\,\rangle \\ \mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 2 - 1) \end{array} \right.$	;
Replace $\triangleright\ \text{if}(\mathsf{F}, \mathsf{x}, \mathsf{y}) \equiv \mathsf{y}\ \triangleright$	$\mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 2 - 1)$	;
Replace $\triangleright\ 2 - 1 \equiv 1\ \triangleright$	$\mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 1)$	;
Definition $\triangleright$	$\mathsf{x} :: \left(1 = 0 \left\{ \begin{array}{l} \langle\,\rangle \\ \mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 1 - 1) \end{array} \right. \right)$	;
Replace $\triangleright\ 1 = 0 \equiv \mathsf{F}\ \triangleright$	$\mathsf{x} :: \left(\mathsf{F} \left\{ \begin{array}{l} \langle\,\rangle \\ \mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 1 - 1) \end{array} \right. \right)$	;
Replace $\triangleright\ \text{if}(\mathsf{F}, \mathsf{x}, \mathsf{y}) \equiv \mathsf{y}\ \triangleright$	$\mathsf{x} :: \mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 1 - 1)$	;
Replace $\triangleright\ 1 - 1 \equiv 0\ \triangleright$	$\mathsf{x} :: \mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 0)$	;
Definition $\triangleright$	$\mathsf{x} :: \mathsf{x} :: \left(0 = 0 \left\{ \begin{array}{l} \langle\,\rangle \\ \mathsf{x} :: (\mathsf{x}\ \text{repeat}\ 0 - 1) \end{array} \right. \right)$	;
Replace $\triangleright\ 0 = 0 \equiv \mathsf{T}\ \triangleright$	$\mathsf{x} :: \mathsf{x} :: \left(\mathsf{T} \left\{ \begin{array}{l} \langle\,\rangle \\ (\mathsf{x} :: \mathsf{x}\ \text{repeat}\ 0 - 1) \end{array} \right. \right)$	;
Replace $\triangleright\ \text{if}(\mathsf{T}, \mathsf{x}, \mathsf{y}) \equiv \mathsf{x}\ \triangleright$	$\mathsf{x} :: \mathsf{x} :: \langle\,\rangle$	;
Reflexivity $\triangleright$	$\langle \mathsf{x}, \mathsf{x} \rangle$	]

In the proof above, I have used the following five computation axioms:

$$\begin{array}{lcl} [\ 2 = 0 & \equiv & \mathsf{F}\ ]^{\cdot} \\ [\ 2 - 1 & \equiv & 1\ ]^{\cdot} \\ [\ 1 = 0 & \equiv & \mathsf{F}\ ]^{\cdot} \\ [\ 1 - 1 & \equiv & 0\ ]^{\cdot} \\ [\ 0 = 0 & \equiv & \mathsf{T}\ ]^{\cdot} \end{array}$$

6.17 Answers

Answer 6.2.1 (a), (b), (c), (p), and (s) are replacements of $[\ (\mathsf{x} :: \mathsf{y})\ \text{tail} \equiv \mathsf{y}\ ]^{\cdot}$.
Answer 6.3.1 (b), (d) and (e) are reverse replacements of $[\ \text{if}(\mathsf{F}, \mathsf{x}, \mathsf{y}) \equiv \mathsf{y}\ ]^{\cdot}$. To see that (e) is a reverse replacement you may proceed as follows: Look up

$$\left[\mathsf{x} \left\{ \begin{array}{l} \mathsf{y} \\ \mathsf{z} \end{array} \right. \right]^{\cdot}$$

in the index. The index points you to Section 2.3. In that section, look for the definition of

$$\left[\mathsf{x}\left\{\begin{array}{c}\mathsf{y}\\ \mathsf{z}\end{array}\right\}\right].$$

The definition reads

$$\left[\mathsf{x}\left\{\begin{array}{c}\mathsf{y}\\ \mathsf{z}\end{array}\right\} \doteq \mathrm{if}(\mathsf{x},\mathsf{y},\mathsf{z})\right].$$

Now count the dots over the equal sign in the definition. There are two dots, so the definition is a macro definition. Therefore,

$$\left[\mathsf{x}\left\{\begin{array}{c}\mathsf{y}\\ \mathsf{z}\end{array}\right\}\right]$$

reduces to $\left[\,\mathrm{if}(\mathsf{x},\mathsf{y},\mathsf{z})\,\right]$ before you have the syntax tree. Once you have the syntax tree it is trivial that (e) is a reverse replacement of $\left[\,\mathrm{if}(\mathsf{F},\mathsf{x},\mathsf{y}) \equiv \mathsf{y}\,\right]$. If there had only been one dot over the equal sign then

$$\left[\mathsf{x}\left\{\begin{array}{c}\mathsf{y}\\ \mathsf{z}\end{array}\right\}\right]$$

would have been a new construct and the syntax tree of (e) would have been different from the syntax tree of $\left[\mathsf{y} \equiv \mathsf{F}\left\{\begin{array}{c}\mathsf{x}\\ \mathsf{y}\end{array}\right\}\right]$. In that case, (e) would not have been a replacement. So the answer is: (e) is a reverse replacement because the definition of

$$\left[\mathsf{x}\left\{\begin{array}{c}\mathsf{y}\\ \mathsf{z}\end{array}\right\}\right]$$

in Section 2.3 has two rather than one dot over the equal sign. If you are a careless reader who don't notice and remember the number of dots over equal signs, you will need to use the index occasionally.

Answer 6.13.5

[Mac proof of L6.13.6**:**

Algebra ▷	$\langle 2,3\rangle$ tail tail	;
Reflexivity ▷	$(2 :: \langle 3\rangle)$ tail tail	;
Replace ▷ $(\mathsf{x} :: \mathsf{y})$ tail $\equiv \mathsf{y}$ ▷	$\langle 3\rangle$ tail	;
Reflexivity ▷	$(3 :: \langle\,\rangle)$ tail	;
Replace ▷ $(\mathsf{x} :: \mathsf{y})$ tail $\equiv \mathsf{y}$ ▷	$\langle\,\rangle$	]

Answer 6.14.3

[Mac proof of L6.14.4**:**

Algebra ▷	$\langle\rangle$ two	;
Definition ▷	$\langle\rangle$ atom$\begin{cases} 2 \\ \langle\rangle \text{ tail two} \end{cases}$	;
Replace ▷ $\langle\rangle$ atom ≡ T ▷	$\mathsf{T}\begin{cases} 2 \\ \langle\rangle \text{ tail two} \end{cases}$	;
Replace ▷ if(T, x, y) ≡ x ▷	2	]

Chapter 7

Variables

7.1 Substitution

You should think of $[\![2 + \mathsf{x}]\!]$ as something that has a value for each value of $[\![\mathsf{x}]\!]$. As examples, if $[\![\mathsf{x}]\!]$ is $[\![5]\!]$ then $[\![2 + \mathsf{x}]\!]$ is $[\![7]\!]$ and if $[\![\mathsf{x}]\!]$ is $[\![\mathsf{T}]\!]$ then $[\![2 + \mathsf{x}]\!]$ is $[\![\bullet]\!]$.

I will use $\left[\!\left[\boxed{\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B} \rangle} \right]\!\right]$ to denote the value of $[\![\mathcal{A}]\!]$ when $[\![\mathsf{x}]\!]$ is $[\![\mathcal{B}]\!]$. As examples, you have

$$[\![\langle 2 + \mathsf{x} \mid \mathsf{x}{:=}5 \rangle \equiv 2 + 5 \equiv 7]\!],$$

$$[\![\langle 2 + \mathsf{x} \mid \mathsf{x}{:=}\mathsf{T} \rangle \equiv 2 + \mathsf{T} \equiv \bullet]\!], \text{ and}$$

$$[\![\langle 2 + \mathsf{x} \mid \mathsf{x}{:=}3 \cdot 5 \rangle \equiv 2 + 3 \cdot 5 \equiv 17]\!].$$

I will refer to the step $[\![\langle 2 + \mathsf{x} \mid \mathsf{x}{:=}5 \rangle \equiv 2 + 5]\!]$ as a *substitution*, and I will say that "$[\![\mathsf{x}]\!]$ is substituted by $[\![5]\!]$ in $[\![2 + \mathsf{x}]\!]$". Here you have examples with several instances of the substitution variable:

$$[\![\langle \mathsf{x} + 2 + \mathsf{x} \mid \mathsf{x}{:=}3 \rangle \equiv 3 + 2 + 3]\!], \text{ and}$$

$$[\![\langle \mathsf{x} + \mathsf{y} + \mathsf{x} + \mathsf{x} \mid \mathsf{x}{:=}3 \cdot 5 \rangle \equiv 3 \cdot 5 + \mathsf{y} + 3 \cdot 5 + 3 \cdot 5]\!].$$

Here you have an example with no instances of the substitution variable:

$$[\![\langle \mathsf{x} + 2 + \mathsf{x} \mid \mathsf{z}{:=}3 \cdot 5 \rangle \equiv \mathsf{x} + 2 + \mathsf{x}]\!].$$

Exercise 7.1.1 Compute the values of the following.

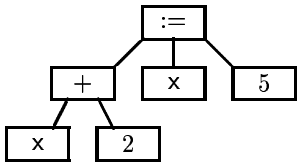
a $[\![\langle \mathsf{y} + 2 + \mathsf{y} \mid \mathsf{y}{:=}4 \rangle]\!]$.

b $[\![\langle 12 \mid \mathsf{x}{:=}13 \rangle]\!]$.

$[\![\langle \mathsf{x} \mid \mathsf{y}{:=}\mathsf{z} \rangle]\!]$ substitute x where y is z end substitution

7.2 How to read substitutions

The syntax tree of $[\,\langle \mathsf{x}+2 \mid \mathsf{x}{:=}5\rangle\,]^{\cdot}$ reads

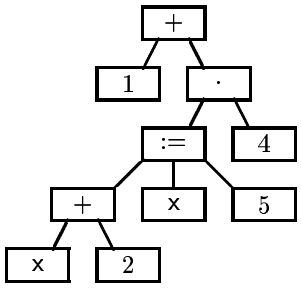


The substitution construct consists of five characters and three subexpressions. The characters are a left elbow $[\,\langle\,]^{\circ}$, a bar $[\,|\,]^{\circ}$, a colon $[\,:\,]^{\circ}$, an equal sign $[\,=\,]^{\circ}$ and a right elbow $[\,\rangle\,]^{\circ}$. Note that the elbows are part of the substitution and delimits the substitution. Map would complain about $[\,\mathsf{x}+2 \mid \mathsf{x} := 5\,]^{\circ}$ because of missing elbows if you replaced the circle superscript by a dot.

The syntax tree of

$$[\,1+\langle \mathsf{x}+2 \mid \mathsf{x}{:=}5\rangle \cdot 4\,]^{\cdot}$$

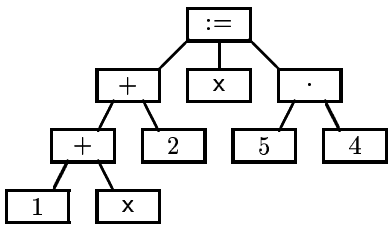
reads



In contrast, the syntax tree of

$$[\,\langle 1+\mathsf{x}+2 \mid \mathsf{x}{:=}5\cdot 4\rangle\,]^{\cdot}$$

reads



This shows that the position of the elbows in the expression does matter.

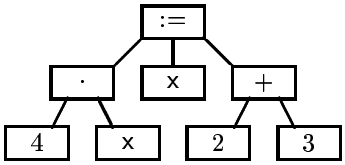
Exercise 7.2.1 Draw the syntax tree of $[\,2\cdot\langle 3+4 \mid \mathsf{x}{:=}5\rangle+6\,]^{\cdot}$.

7.3 Only variables can be substituted

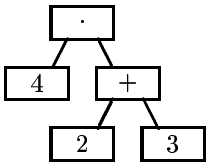
$[\,\langle \mathcal{A} \mid \mathcal{B}{:=}\mathcal{C}\rangle\,]^{\cdot}$ is special in that $[\,\langle \mathcal{A} \mid \mathcal{B}{:=}\mathcal{C}\rangle\,]^{\cdot}$ only makes sense when $[\,\mathcal{B}\,]^{\cdot}$ is a variable. As an example, $[\,\langle 2+3 \mid 2{:=}5\rangle\,]^{\circ}$ does not make sense. If you replace the circle superscript by a dot, then Map will complain and say that $[\,\langle 2+3 \mid 2{:=}5\rangle\,]^{\circ}$ is syntactically invalid.

7.4 Substitution in syntax trees

If you textually replace $[\,\mathsf{x}\,]^{\cdot}$ by $[\,2+3\,]^{\cdot}$ in $[\,4\cdot\mathsf{x}\,]^{\cdot}$ then you get $[\,4\cdot 2+3\,]^{\cdot}$ which means $[\,(4\cdot 2)+3\,]^{\cdot}$. However, $[\,\langle 4\cdot\mathsf{x} \mid \mathsf{x}{:=}2+3\rangle\,]^{\cdot}$ does not ask for textual replacement. Rather, $[\,\langle 4\cdot\mathsf{x} \mid \mathsf{x}{:=}2+3\rangle \equiv 4\cdot(2+3)\,]^{\cdot}$. Where did the parenthesis come from? This is obvious if you look at syntax trees. The syntax tree of $[\,\langle 4\cdot\mathsf{x} \mid \mathsf{x}{:=}2+3\rangle\,]^{\cdot}$ reads



When you perform the substitution, you should replace each $[\,\mathsf{x}\,]^{\cdot}$ in the first subtree by the third subtree. This gives



You can then express the syntax tree as $[\,4\cdot(2+3)\,]^{\cdot}$. As you can see, the parenthesis came from the translation from syntax tree to term.

At least mentally, you should always perform substitutions on syntax trees to get the parentheses right.

Exercise 7.4.1 Compute $[\,\langle \mathsf{x}\cdot\mathsf{x} \mid \mathsf{x}{:=}5-2\rangle\,]^{\cdot}$.

7.5 Algebraic rules for substitution

Take a look at this:

$$\begin{array}{lcl} \langle \mathsf{x}\cdot 3+\mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle & \equiv & 2\cdot 3+2\cdot\mathsf{y} \\ \langle \mathsf{x}\cdot 3 \mid \mathsf{x}{:=}2\rangle & \equiv & 2\cdot 3 \\ \langle \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle & \equiv & 2\cdot\mathsf{y} \end{array}$$

As you can see, you have

$$[\,\langle \mathsf{x}\cdot 3+\mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle \equiv \langle \mathsf{x}\cdot 3 \mid \mathsf{x}{:=}2\rangle + \langle \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle\,]^{\cdot}.$$

In general, you have

[**Mac rule SubXPlusY** : $\langle \mathcal{A} + \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv \langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S} \rangle + \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S} \rangle$]

In other words, when you replace $[\,\mathsf{x}\,]^{\cdot}$ by $[\,\mathcal{S}\,]^{\cdot}$ in $[\,\mathcal{A} + \mathcal{B}\,]^{\cdot}$, then you do it by replacing $[\,\mathsf{x}\,]^{\cdot}$ by $[\,\mathcal{S}\,]^{\cdot}$ in $[\,\mathcal{A}\,]^{\cdot}$ and $[\,\mathcal{B}\,]^{\cdot}$ and adding the results.

$[\,\langle \mathcal{A} + \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv \langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S} \rangle + \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S} \rangle\,]^{\cdot}$ is one in a family of algebraic rules that describe substitution. Here are some more:

[**Mac rule SubXTimesY** : $\langle \mathcal{A} \cdot \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv \langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S} \rangle \cdot \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S} \rangle$]

[**Mac rule SubIf** : $\langle \mathrm{if}(\mathcal{A}, \mathcal{B}, \mathcal{C}) \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv$
$\mathrm{if}(\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S} \rangle, \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S} \rangle, \langle \mathcal{C} \mid \mathsf{x}{:=}\mathcal{S} \rangle)$]

[**Mac rule SubT** : $\langle \mathsf{T} \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv \mathsf{T}$]

[**Mac rule SubFThreeOfXEnd** : $\langle f_3(\mathcal{A}) \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv f_3(\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S} \rangle)$]

7.6 Substitution and numerals

Numerals are constructs that consist of sequences of digits, possibly with a decimal point $[\,.\,]^{\circ}$ and a floating point mark $[\,\mathtt{F}\,]^{\circ}$. $[\,117\,]^{\cdot}$ and $[\,1.234\mathtt{F}\,]^{\cdot}$ are numerals. $[\,\infty\,]^{\cdot}$ and $[\,2 + 3\,]^{\cdot}$ are not numerals.

You have

$[\,\langle 2 \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv 2\,]^{\cdot}$, and

$[\,\langle 1.23\mathtt{F} \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv 1.23\mathtt{F}\,]^{\cdot}$.

In general, you have

$[\,\langle \mathsf{y} \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv \mathsf{y}\,]^{\cdot}$

for all numerals $[\,\mathsf{y}\,]^{\cdot}$. I will state this thus:

[**Mac rule SubNumeral** : **if** numeral(y) **then** $\langle \mathsf{y} \mid \mathsf{x}{:=}\mathcal{S} \rangle \equiv \mathsf{y}$]

I will refer to $[\,\mathrm{numeral}(\mathsf{y})\,]^{\cdot}$ as a *side condition*. Side conditions say something about the shape of their arguments. As examples, $[\,\mathrm{numeral}(4)\,]^{\cdot}$ holds and $[\,\mathrm{numeral}(2 + 2)\,]^{\cdot}$ fails. Side conditions do not respect substitution of equals. I return to side conditions later.

numeral
side condition

7.7 Substitution and variables

What is $[\ \langle \mathsf{x} \mid \mathsf{x}{:=}\mathcal{S}\rangle\ ]$? Well, to find $[\ \langle \mathsf{x} \mid \mathsf{x}{:=}\mathcal{S}\rangle\ ]$ you have to replace $[\ \mathsf{x}\ ]$ by $[\ \mathcal{S}\ ]$ in the term $[\ \mathsf{x}\ ]$. The result is $[\ \mathcal{S}\ ]$. Hence, you have

$[\ \textbf{Mac rule SubXX}\ :\ \langle \mathsf{x} \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \mathcal{S}\ ]$

If $[\ \mathsf{x}\ ]$ and $[\ \mathsf{y}\ ]$ are distinct variables, then $[\ \mathsf{x}\ ]$ does not occur in the term $[\ \mathsf{y}\ ]$ and you have

$[\ \textbf{Mac rule SubXY}\ :\ \textbf{if}\ \ \mathrm{distinct}(\mathsf{x},\mathsf{y})\ \textbf{then}\ \langle \mathsf{y} \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \mathsf{y}\ ]$

Here you have an algebraic proof of $[\ \langle \mathsf{x}\cdot 3 + \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle \equiv 2\cdot 3 + 2\cdot\mathsf{y}\ ]$:

$[\ \textbf{Mac lemma L7.7.1}\ :\ \langle \mathsf{x}\cdot 3 + \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle \equiv 2\cdot 3 + 2\cdot\mathsf{y}\ ]$

$[\ \textbf{Mac proof of}$ L7.7.1**:**

Algebra $\triangleright$	$\langle \mathsf{x}\cdot 3 + \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle$	;
Replace $\triangleright$ SubXPlusY $\triangleright$	$\langle \mathsf{x}\cdot 3 \mid \mathsf{x}{:=}2\rangle + \langle \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle$	;
Replace $\triangleright$ SubXTimesY $\triangleright$	$\langle \mathsf{x} \mid \mathsf{x}{:=}2\rangle \cdot \langle 3 \mid \mathsf{x}{:=}2\rangle + \langle \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle$	;
Replace $\triangleright$ SubXX $\triangleright$	$2 \cdot \langle 3 \mid \mathsf{x}{:=}2\rangle + \langle \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle$	;
Replace $\triangleright$ SubNumeral $\triangleright$	$2\cdot 3 + \langle \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}2\rangle$	;
Replace $\triangleright$ SubXTimesY $\triangleright$	$2\cdot 3 + \langle \mathsf{x} \mid \mathsf{x}{:=}2\rangle \cdot \langle \mathsf{y} \mid \mathsf{x}{:=}2\rangle$	;
Replace $\triangleright$ SubXX $\triangleright$	$2\cdot 3 + 2\cdot \langle \mathsf{y} \mid \mathsf{x}{:=}2\rangle$	;
Replace $\triangleright$ SubXY $\triangleright$	$2\cdot 3 + 2\cdot\mathsf{y}$	$]$

As you can see, algebraic proofs about substitution are very tedious. It is good to know that such proofs exist. Occasionally you may want to resort to such proofs in complicated situations, and you may program computers to perform substitutions automatically, but you should not write proofs like the one above in your daily work.

Exercise 7.7.2 Prove this:

$[\ \textbf{Mac lemma L7.7.3}\ :\ \langle \mathrm{if}(\mathsf{x},\mathsf{y},\mathsf{z}) \mid \mathsf{x}{:=}\mathsf{T}\rangle \equiv \mathsf{y}\ ]$

7.8 Binding

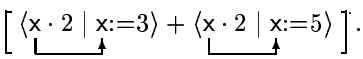
You have

$$[\ \langle \mathsf{x}\cdot 2 \mid \mathsf{x}{:=}3\rangle + \langle \mathsf{x}\cdot 2 \mid \mathsf{x}{:=}5\rangle \equiv 3\cdot 2 + 5\cdot 2\ ].$$

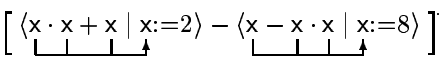
As you can see, there are four occurrences of $[\ \mathsf{x}\ ]$ here. The second occurrence specifies that the first occurrence equals $[\ 3\ ]$ and the fourth occurrence specifies that the third occurrence equals $[\ 5\ ]$. I will say that the second occurrence *binds*

bind

the first occurrence and the fourth binds the third. I will illustrate this by a *binding diagram*:



Here you have another example:



You have that the first, second and third $[\,x\,]$ are bound by the fourth, and the fifth, sixth and seventh are bound by the eighth. The term behaves as if it contains two distinct variables: The first four occurrences of $[\,x\,]$ behave like one variable and the next four behave like another.

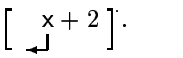
Exercise 7.8.1 Draw binding diagrams for the following:

a $[\,\langle x \cdot x \mid x{:=}2+2\rangle + \langle x+x \mid x{:=}1\rangle\,]$.

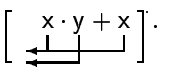
b $[\,\langle x \mid x{:=}2\rangle + \langle x \mid x{:=}3\rangle + \langle x \mid x{:=}4\rangle\,]$.

7.9 Free variables

$[\,x\,]$ in $[\,x+2\,]$ is not bound, so I will call it *free*. In binding diagrams I will illustrate that as follows:



The arrow indicates that $[\,x\,]$ is not bound by anything. Here is a diagram for a term with two free variables, $[\,x\,]$ and $[\,y\,]$:



The term $[\,x+2\,]$ has a value for each value of $[\,x\,]$. If $[\,x\,]$ is $[\,5\,]$ then $[\,x+2\,]$ is $[\,7\,]$ and if $[\,x\,]$ is $[\,\mathsf{T}\,]$ then $[\,x+2\,]$ is $[\,\bullet\,]$. Hence, the value of $[\,x+2\,]$ depends on the value of $[\,x\,]$. A term does not necessarily depend on the values of its free variables. As an example, $[\,\mathrm{if}(\mathsf{T},\mathsf{T},x)\,]$ does not depend on the value of the free variable $[\,x\,]$. On the contrary, a term is definitely independent of any variable that is not free within it.

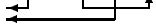
---

binding diagram
free

7.10 Bound versus binding variables

The binding diagram of $[\,x + \langle x \cdot y \mid x{:=}2\rangle\,]$ reads

$$\left[\;\; x + \langle x \cdot y \mid x{:=}2\rangle \;\right]$$



The term contains three occurrences of $[\,x\,]$ and one of $[\,y\,]$. I will say that the first occurrence of $[\,x\,]$ is free, the second is *bound* and the third is *binding*. The sole occurrence of $[\,y\,]$ is free. The literature does not distinguish between bound and binding occurrences.

$[\,\langle 12 \mid x{:=}13\rangle\,]$ has one binding and no bound occurrences of $[\,x\,]$; its binding diagram is

$$\left[\;\langle 12 \mid x{:=}13\rangle\;\right]$$

Exercise 7.10.1 For each occurrence of $[\,x\,]$ and $[\,y\,]$, tell whether the occurrence is free, bound or binding:

$$[\,\langle x + y \mid x{:=}2\rangle + x \cdot x + \langle x \cdot y \mid y{:=}3\rangle\,].$$

7.11 Renaming of bound variables

You have

$$[\,\langle x + 2 \mid x{:=}5\rangle \equiv 5 + 2\,] \text{ and}$$

$$[\,\langle y + 2 \mid y{:=}5\rangle \equiv 5 + 2\,].$$

which proves

$$[\,\langle x + 2 \mid x{:=}5\rangle \equiv \langle y + 2 \mid y{:=}5\rangle\,].$$

The only difference between $[\,\langle x + 2 \mid x{:=}5\rangle\,]$ and $[\,\langle y + 2 \mid y{:=}5\rangle\,]$ is the name of the bound variable.

In general, names of bound variables are immaterial. You can replace $[\,x\,]$ by any other variable in $[\,\langle x+2 \mid x{:=}5\rangle\,]$ without affecting the value of the term.

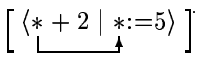
If you draw the binding diagrams of $[\,\langle x + 2 \mid x{:=}5\rangle\,]$ and $[\,\langle y + 2 \mid y{:=}5\rangle\,]$ then you get

$$\left[\;\langle x + 2 \mid x{:=}5\rangle\;\right] \text{ and}$$

$$\left[\;\langle y + 2 \mid y{:=}5\rangle\;\right]$$

bound
binding

If you replace the bound and binding variables by asterisks, then you obtain

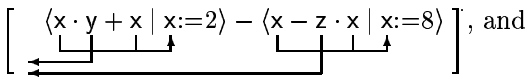


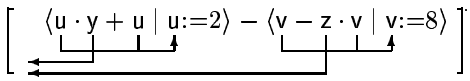
in both cases. I will call this diagram the *anonymous binding diagram* of $\lceil \langle \mathsf{x}+2 \mid \mathsf{x}:=5\rangle \rceil^{\cdot}$ as well as $\lceil \langle \mathsf{y}+2 \mid \mathsf{y}:=5\rangle \rceil^{\cdot}$. In general, you obtain the anonymous binding diagram of a term by replacing all bound and binding variables in the binding diagram by asterisks (free variables are not changed).

If two terms have identical anonymous binding diagrams, then I will say that the terms are *structurally equal* or *identical except for naming of bound variables*.

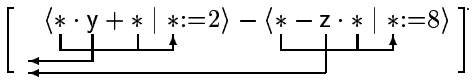
If two terms $\lceil \mathcal{A} \rceil^{\cdot}$ and $\lceil \mathcal{B} \rceil^{\cdot}$ are structurally equal, then $\lceil \mathcal{A} \equiv \mathcal{B} \rceil^{\cdot}$, and you can prove $\lceil \mathcal{A} \equiv \mathcal{B} \rceil^{\cdot}$ by the algebraic rules for substitution. The opposite does not hold. As an example, $\lceil 2+3 \equiv 3+2 \rceil^{\cdot}$, but $\lceil 2+3 \rceil^{\cdot}$ and $\lceil 3+2 \rceil^{\cdot}$ are structurally different.

As an example, the two terms

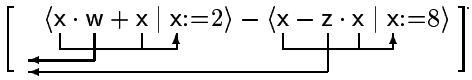
 , and



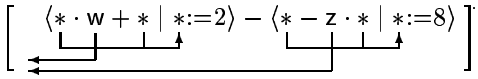
both have anonymous binding diagrams



so they are structurally equal. In contrast,



has anonymous binding diagram



so it is structurally different from the two terms above.

Exercise 7.11.1 Draw anonymous binding diagrams for the following terms and tell which ones are structurally equal.

a $\lceil \langle \mathsf{x}+\mathsf{w} \mid \mathsf{x}:=2\rangle + \langle \mathsf{x}+\mathsf{w} \mid \mathsf{x}:=3\rangle \rceil^{\cdot}$.

b $\lceil \langle \mathsf{x}+\mathsf{w} \mid \mathsf{x}:=2\rangle + \langle \mathsf{y}+\mathsf{v} \mid \mathsf{y}:=3\rangle \rceil^{\cdot}$.

c $\lceil \langle \mathsf{z}+\mathsf{w} \mid \mathsf{z}:=2\rangle + \langle \mathsf{u}+\mathsf{w} \mid \mathsf{u}:=3\rangle \rceil^{\cdot}$

d $\lceil \langle \mathsf{z}+\mathsf{w} \mid \mathsf{z}:=3\rangle + \langle \mathsf{v}+\mathsf{w} \mid \mathsf{v}:=3\rangle \rceil^{\cdot}$

---

anonymous binding diagram
structurally equal
identical except for naming of bound variables

7.12 No renaming of free variables

If $[\![x]\!]$ is $[\![2]\!]$ and $[\![y]\!]$ is $[\![3]\!]$ then $[\![x + 2]\!]$ is $[\![4]\!]$ and $[\![y + 2]\!]$ is $[\![5]\!]$. This shows that $[\![x + 2]\!]$ and $[\![y + 2]\!]$ are not equal in general. Hence, in general, you cannot rename free variables in a term without affecting the term.

7.13 Variable clashes

$[\![\langle u \cdot x \mid u{:=}2 \rangle]\!]$ is structurally equal to $[\![\langle v \cdot x \mid v{:=}2 \rangle]\!]$. With one exception, you can replace $[\![u]\!]$ with any other variable without affecting the value of the term. The exception is important: You cannot replace $[\![u]\!]$ by $[\![x]\!]$ without affecting the value. If you replace $[\![u]\!]$ by $[\![x]\!]$ then you obtain $[\![\langle x \cdot x \mid x{:=}2 \rangle]\!]$, but $[\![\langle x \cdot x \mid x{:=}2 \rangle \equiv 4]\!]$ whereas $[\![\langle u \cdot x \mid u{:=}2 \rangle \equiv 2 \cdot x]\!]$.

When you rename a bound variable you have to make sure that the term remains structurally unchanged. Renaming of $[\![u]\!]$ to $[\![x]\!]$ changes the structure from

$$[\![\quad \langle u \cdot x \mid u{:=}2 \rangle \quad]\!] \text{ to}$$

$$[\![\quad \langle x \cdot x \mid x{:=}2 \rangle \quad]\!].$$

Exercise 7.13.1 Draw a binding diagram for $[\![\langle x \cdot v \mid x{:=}2 \rangle + \langle y + z \cdot u \mid y{:=}3 \rangle]\!]$. What variables can $[\![x]\!]$ be replaced by without affecting the term? Same question for $[\![y]\!]$.

7.14 Nested substitution

What is the value of $[\![\langle \langle x + y \mid x{:=}2 \rangle \mid y{:=}3 \rangle]\!]$? Well, let us try to compute it. $[\![\langle x + y \mid x{:=}2 \rangle \equiv 2 + y]\!]$ so you have

$$\begin{array}{ll} [\![& \langle \langle x + y \mid x{:=}2 \rangle \mid y{:=}3 \rangle \equiv \\ & \langle 2 + y \mid y{:=}3 \rangle \equiv \\ & 2 + 3 \equiv \\ & 5 \,]\!] \end{array}$$

This is not particularly surprising: The starting point is $[\![x + y]\!]$, then $[\![x]\!]$ is replaced by $[\![2]\!]$ and $[\![y]\!]$ by $[\![3]\!]$ so the result of the substitutions is $[\![2 + 3]\!]$ which in turn equals $[\![5]\!]$.

This one is more complicated:

$$\begin{array}{ll} [\![& \langle \langle x + y \mid x{:=}2 \cdot y \rangle \mid y{:=}3 \rangle \equiv \\ & \langle 2 \cdot y + y \mid y{:=}3 \rangle \equiv \\ & 2 \cdot 3 + 3 \,]\!] \end{array}$$

As you can see, $[\![x]\!]$ is replaced by $[\![2 \cdot y]\!]$, and then $[\![y]\!]$ is replaced by $[\![3]\!]$.

This one is even more tricky:

$$\begin{array}{ll}[& \langle\langle \mathsf{x}+\mathsf{y} \mid \mathsf{x}{:=}2\rangle \mid \mathsf{x}{:=}3\rangle \equiv \\ & \langle 2+\mathsf{y} \mid \mathsf{x}{:=}3\rangle \equiv \\ & 2+\mathsf{y}\ ]^{\cdot}\end{array}$$

As you can see, $[\,\mathsf{x}\,]^{\cdot}$ is replaced by $[\,2\,]^{\cdot}$, and then there are no occurrences of $[\,\mathsf{x}\,]^{\cdot}$ left to replace by $[\,3\,]^{\cdot}$ afterwards.

This is really mean:

$$\begin{array}{ll}[& \langle\langle \mathsf{x}+\mathsf{y} \mid \mathsf{x}{:=}\mathsf{x}\cdot 2\rangle \mid \mathsf{x}{:=}4\rangle \equiv \\ & \langle \mathsf{x}\cdot 2+\mathsf{y} \mid \mathsf{x}{:=}4\rangle \equiv \\ & 4\cdot 2+\mathsf{y}\ ]^{\cdot}\end{array}$$

As you can see, $[\,\mathsf{x}\,]^{\cdot}$ is replaced by something that contains $[\,\mathsf{x}\,]^{\cdot}$, and then that $[\,\mathsf{x}\,]^{\cdot}$ is replaced by $[\,4\,]^{\cdot}$.

Exercise 7.14.1 Reduce the following as much as possible:

a $[\,\langle\langle\langle \mathsf{x}+\mathsf{y} \mid \mathsf{x}{:=}\mathsf{x}\cdot\mathsf{y}\rangle \mid \mathsf{y}{:=}\mathsf{x}\rangle \mid \mathsf{x}{:=}2\rangle\,]^{\cdot}$.

b $[\,\langle \mathsf{x}\cdot\mathsf{y} \mid \mathsf{x}{:=}\langle \mathsf{y}+\mathsf{x} \mid \mathsf{y}{:=}2\rangle\rangle\,]^{\cdot}$.

c $[\,\langle\langle \mathsf{x}+\mathsf{y} \mid \mathsf{x}{:=}3\rangle+\mathsf{y} \mid \mathsf{y}{:=}2\rangle\,]^{\cdot}$.

7.15 Binding and nested substitution

In $[\,\langle \mathsf{x}+2 \mid \mathsf{x}{:=}\mathsf{x}\cdot 3\rangle\,]^{\cdot}$ the first $[\,\mathsf{x}\,]^{\cdot}$ is bound and the second is binding, but what about the third $[\,\mathsf{x}\,]^{\cdot}$? You have $[\,\langle \mathsf{x}+2 \mid \mathsf{x}{:=}\mathsf{x}\cdot 3\rangle \equiv \mathsf{x}\cdot 3+2\,]^{\cdot}$. If $[\,\mathsf{x}\,]^{\cdot}$ is $[\,5\,]^{\cdot}$ then $[\,\langle \mathsf{x}+2 \mid \mathsf{x}{:=}\mathsf{x}\cdot 3\rangle \equiv 5\cdot 3+2 \equiv 17\,]^{\cdot}$ and is $[\,\mathsf{x}\,]^{\cdot}$ is $[\,10\,]^{\cdot}$ then $[\,\langle \mathsf{x}+2 \mid \mathsf{x}{:=}\mathsf{x}\cdot 3\rangle \equiv 10\cdot 3+2 \equiv 32\,]^{\cdot}$. This shows that $[\,\langle \mathsf{x}+2 \mid \mathsf{x}{:=}\mathsf{x}\cdot 3\rangle\,]^{\cdot}$ depends on the value of the third $[\,\mathsf{x}\,]^{\cdot}$, so the third $[\,\mathsf{x}\,]^{\cdot}$ is free. Hence, the binding diagram looks as follows:

$$\left[\ \langle \mathsf{x}+2 \mid \mathsf{x}{:=}\mathsf{x}\cdot 3\rangle\ \right]^{\cdot}.$$

In general you have that the $[\,\mathsf{x}\,]^{\cdot}$ in $[\,\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle\,]^{\cdot}$ binds all occurrences of $[\,\mathsf{x}\,]^{\cdot}$ that are free in $[\,\mathcal{A}\,]^{\cdot}$ but does not bind any occurrences of $[\,\mathsf{x}\,]^{\cdot}$ in $[\,\mathcal{B}\,]^{\cdot}$. In Section A.28 I have stated the binding properties of $[\,\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle\,]^{\cdot}$ thus:

$$\left[\ \langle * \mid *{:=}*\rangle\ \right]^{\cdot}.$$

Here is a more complicated example of nested substitution:

$$\left[\ \langle\langle \mathsf{x}+2 \mid \mathsf{x}{:=}\mathsf{x}\cdot 3\rangle \mid \mathsf{x}{:=}\mathsf{x}-1\rangle\ \right]^{\cdot}.$$

If you perform the substitutions you obtain

$$\begin{array}{ll}[& \langle\langle x + 2 \mid x{:=}x \cdot 3\rangle \mid x{:=}x - 1\rangle \equiv \\ & \langle x \cdot 3 + 2 \mid x{:=}x - 1\rangle \equiv \\ & (x - 1) \cdot 3 + 2\ ]\end{array}$$

If you rename the bound variables, then the structure becomes clearer:

$$[\ \langle\langle u + 2 \mid u{:=}v \cdot 3\rangle \mid v{:=}x - 1\rangle\ ].$$

In this case the substitution reads

$$\begin{array}{l}[\ \langle\langle u + 2 \mid u{:=}v \cdot 3\rangle \mid v{:=}x - 1\rangle \equiv \\ \langle v \cdot 3 + 2 \mid v{:=}x - 1\rangle \equiv \\ (x - 1) \cdot 3 + 2\ ]\end{array}$$

I will say that a term is *simple* if any two binding variables are distinct and any binding variable is distinct from any free one. As an example, $[\ \langle\langle u + 2 \mid u{:=}v \cdot 3\rangle \mid v{:=}x - 1\rangle\ ]$ is simple and $[\ \langle\langle x + 2 \mid x{:=}x \cdot 3\rangle \mid x{:=}x - 1\rangle\ ]$ is not.

I will say that you *simplify* a term if you rename bound variables such that the result is a simple term. You can simplify any term by suitable renaming of bound variables. As an example,

$$\langle\langle x \cdot x \mid x{:=}x + 1\rangle + x \mid x{:=}x^2\rangle$$

can be simplified into

$$[\ \langle\langle y \cdot y \mid y{:=}z + 1\rangle + z \mid z{:=}x^2\rangle\ ].$$

Exercise 7.15.1 Simplify the following terms:

a $[\ \langle\langle x \cdot y \mid x{:=}y\rangle \mid y{:=}x + \langle x \cdot y \mid x{:=}5\rangle\rangle\ ]$.

b $[\ \langle\langle x - 1 \mid x{:=}x \cdot x\rangle + x \mid x{:=}x - 1\rangle\ ]$.

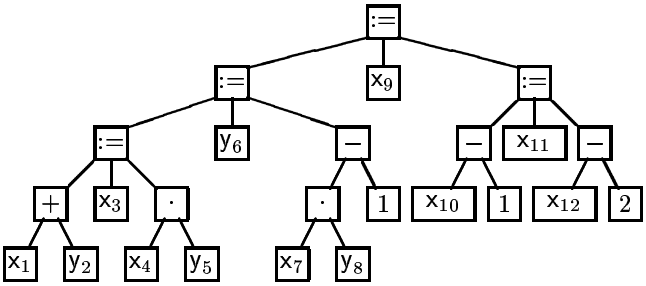
7.16 Binding diagrams and syntax trees

The syntax tree of

$$[\ \langle\langle\langle x + y \mid x{:=}x \cdot y\rangle \mid y{:=}x \cdot y - 1\rangle \mid x{:=}\langle x - 1 \mid x{:=}x - 2\rangle\rangle\ ]$$

is

simple term
simplify a term



where I have taken the liberty to put a subscript on each occurrence of a variable.

You can find the binding arrows of the variables as follows: Put your finger on the occurrence in question. Then move upwards in the tree. Each time you meet a $[\,:=\,]^\circ$ node along the rightmost edge, then continue up the tree. If you reach a $[\,:=\,]^\circ$ node along the middle edge, then the variable is binding.

If you reach a $[\,:=\,]^\circ$ node along the leftmost edge, then do as follows: If the variable in question equals the binding variable of the $[\,:=\,]^\circ$ node, then the variable in question is bound by that binding variable. If the variable in question is distinct from the binding variable, then continue up the tree.

If you pop out on top of the tree, then the variable in question is free.

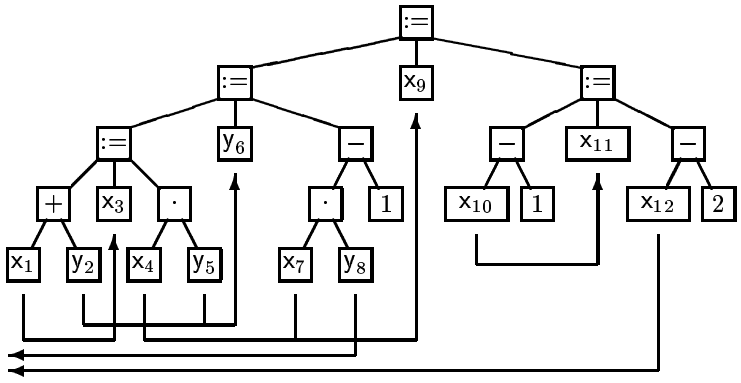
Example. If you move up from $[\,x_3\,]^\cdot$, you reach a $[\,:=\,]^\circ$ node along the middle edge so $[\,x_3\,]^\cdot$ is binding.

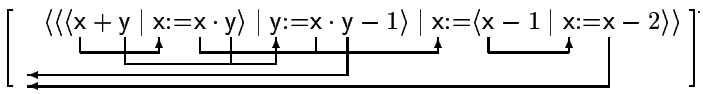
Example. If you move up from $[\,y_2\,]^\cdot$ then you first reach a $[\,+\,]^\circ$ node and then a $[\,:=\,]^\circ$ node. The binding variable of the $[\,:=\,]^\circ$ node is $[\,x_3\,]^\cdot$. Since $[\,x\,]^\cdot$ and $[\,y\,]^\cdot$ are distinct, you have to continue upwards. You then reach a $[\,:=\,]^\circ$ node whose binding variable is $[\,y_6\,]^\cdot$, so $[\,y_2\,]^\cdot$ is bound by $[\,y_6\,]^\cdot$.

Example. If you move up from $[\,x_4\,]^\cdot$, then you first reach a $[\,\cdot\,]^\circ$ node, then a $[\,:=\,]^\circ$ node along the rightmost edge, then a $[\,:=\,]^\circ$ node with a distinct binding variable, and finally a $[\,:=\,]^\circ$ node with binding variable $[\,x_9\,]^\cdot$. Hence, $[\,x_4\,]^\cdot$ is bound by $[\,x_9\,]^\cdot$.

Example. If you move up from $[\,x_{12}\,]^\cdot$, then you first reach a $[\,-\,]^\circ$ node, then a $[\,:=\,]^\circ$ node along the rightmost edge, then another $[\,:=\,]^\circ$ node along the rightmost edge and then you pop out on top of the tree. Hence, $[\,x_{12}\,]^\cdot$ is free.

The binding arrows are as follows:





Exercise 7.16.1 Draw the syntax tree and binding diagram for the term

$$[\,\langle\langle x + y \mid y{:=}x \cdot y\rangle + y \mid x{:=}\langle x - y \mid x{:=}2 \cdot x\rangle\rangle \mid x{:=}\langle z - x \mid z{:=}u - v\rangle\rangle\,]^{\cdot}.$$

7.17 Substitution inside out

You have

$$\begin{array}{ll}[& \langle\langle x + 2 \mid x{:=}3\rangle \mid x{:=}4\rangle \equiv \\ & \langle 3 + 2 \mid x{:=}4\rangle \equiv \\ & 3 + 2\,]^{\cdot}\end{array}$$

In the first step I replace $[\,\langle x + 2 \mid x{:=}3\rangle\,]^{\cdot}$ with $[\,3 + 2\,]^{\cdot}$ which is legal because $[\,\langle x + 2 \mid x{:=}3\rangle \equiv 3 + 2\,]^{\cdot}$. This is an example of substitution of equals, which I mentioned in Section 1.5.

This, however, is wrong:

$$\begin{array}{l}\langle\langle x + 2 \mid x{:=}3\rangle \mid x{:=}4\rangle \equiv \\ \langle 4 + 2 \mid x{:=}3\rangle \equiv \\ 4 + 2\end{array}$$

In the first step I replace $[\,x\,]^{\cdot}$ by $[\,4\,]^{\cdot}$ using the outermost substitution, but that is illegal. The binding diagram

clearly shows that $[\,x\,]^{\cdot}$ must be replaced by $[\,3\,]^{\cdot}$. This is even worse:

$$[\,\langle\langle x + 2 \mid x{:=}3\rangle \mid x{:=}4\rangle \equiv \langle 4 + 2 \mid 4{:=}3\rangle\,]^{\circ}.$$

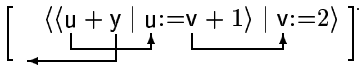
This is syntactically invalid because $[\,\mathcal{B}\,]^{\cdot}$ in $[\,\langle\mathcal{A} \mid \mathcal{B}{:=}\mathcal{C}\rangle\,]^{\cdot}$ must be a variable.

7.18 Substitution outside in

Consider the expression

$$[\,\langle\langle x + y \mid x{:=}x + 1\rangle \mid x{:=}2\rangle\,]^{\cdot}.$$

Suppose that you, for some reason, want to perform the outermost substitution first. The safest thing to do is to simplify the term first:



As soon as the term is simple, you can perform the outermost substitution simply by replacing all occurrences of $[\,\mathsf{v}\,]$ by $[\,2\,]$:

$$[\,\langle \mathsf{u}+\mathsf{y} \mid \mathsf{u}{:=}2+1\rangle\,].$$

If you just replace all occurrences of $[\,\mathsf{x}\,]$ by $[\,2\,]$ in $[\,\langle\langle \mathsf{x}+\mathsf{y} \mid \mathsf{x}{:=}\mathsf{x}+1\rangle \mid \mathsf{x}{:=}2\rangle\,]$ then you obtain

$$[\,\langle 2+\mathsf{y} \mid 2{:=}2+1\rangle\,]^{\circ}$$

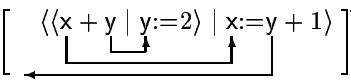
which is syntactically invalid since $[\,\mathcal{B}\,]$ in $[\,\langle \mathcal{A} \mid \mathcal{B}{:=}\mathcal{C}\rangle\,]$ must be a variable. If you don't replace binding occurrences of $[\,\mathsf{x}\,]$ you get

$$[\,\langle 2+\mathsf{y} \mid \mathsf{x}{:=}2+1\rangle\,]$$

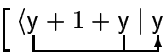
which is syntactically valid but still not correct. If you really want to perform the outermost substitution without simplifying the term first, then you should only substitute those occurrences of $[\,\mathsf{x}\,]$ that are free in $[\,\langle \mathsf{x}+\mathsf{y} \mid \mathsf{x}{:=}\mathsf{x}+1\rangle\,]$:

$$[\,\langle\langle \mathsf{x}+\mathsf{y} \mid \mathsf{x}{:=}\mathsf{x}+1\rangle \mid \mathsf{x}{:=}2\rangle \equiv \langle \mathsf{x}+\mathsf{y} \mid \mathsf{x}{:=}2+1\rangle\,].$$

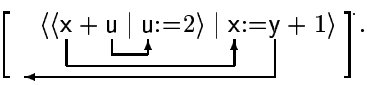
Here is a more complicated example:



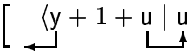
If you just replace $[\,\mathsf{x}\,]$ by $[\,\mathsf{y}+1\,]$ you incorrectly obtain



The problem is that the free variable $[\,\mathsf{y}\,]$ is caught by the binding $[\,\mathsf{y}\,]$. You may simplify the expression into



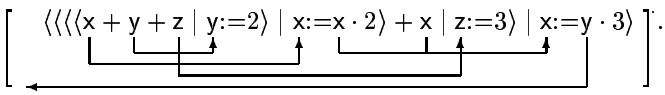
and then replace $[\,\mathsf{x}\,]$ by $[\,\mathsf{y}+1\,]$:



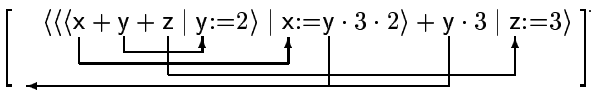
The general rule is as follows:

> Suppose $[\,\mathcal{A}\,]$ and $[\,\mathcal{B}\,]$ are terms and $[\,\mathsf{x}\,]$ is a variable. Suppose $[\,\mathcal{A}'\,]$ arises from $[\,\mathcal{A}\,]$ by replacing all free occurrences of $[\,\mathsf{x}\,]$ in $[\,\mathcal{A}\,]$ by $[\,\mathcal{B}\,]$. Further suppose that no free variable in $[\,\mathcal{B}\,]$ becomes bound in $[\,\mathcal{A}'\,]$. In this case, $[\,\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle \equiv \mathcal{A}'\,]$.

The condition that no free variable in $[\![\ \mathcal{B}\ ]\!]$ becomes bound in $[\![\ \mathcal{A}'\ ]\!]$ is always met if $[\![\ \langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle\ ]\!]$ is simple, but is also met in many other cases. As an example, consider



Here you can replace the two occurrences of $[\![\ \mathsf{x}\ ]\!]$ that are bound in the outermost construct by $[\![\ \mathsf{y}\cdot 3\ ]\!]$ since the $[\![\ \mathsf{y}\ ]\!]$ in $[\![\ \mathsf{y}\cdot 3\ ]\!]$ does not become bound:



The literature says that $[\![\ \mathcal{B}\ ]\!]$ is *free for* $[\![\ \mathsf{x}\ ]\!]$ in $[\![\ \mathcal{A}\ ]\!]$ if no free variable of $[\![\ \mathcal{B}\ ]\!]$ becomes bound when all free occurrences of $[\![\ \mathsf{x}\ ]\!]$ in $[\![\ \mathcal{A}\ ]\!]$ are replaced by $[\![\ \mathcal{B}\ ]\!]$. Hence, the rule above can also be stated:

> $[\![\ \langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle \equiv \mathcal{A}'\ ]\!]$ if $[\![\ \mathcal{A}'\ ]\!]$ arises from $[\![\ \mathcal{A}\ ]\!]$ be replacing all free occurrences of $[\![\ \mathsf{x}\ ]\!]$ in $[\![\ \mathcal{A}\ ]\!]$ by $[\![\ \mathcal{B}\ ]\!]$ and if $[\![\ \mathcal{B}\ ]\!]$ is free for $[\![\ \mathsf{x}\ ]\!]$ in $[\![\ \mathcal{A}\ ]\!]$.

Exercise 7.18.1 Is $[\![\ \mathcal{B}\ ]\!]$ free for $[\![\ \mathsf{x}\ ]\!]$ in $[\![\ \mathcal{A}\ ]\!]$ in the following cases?

a $[\![\ \mathcal{A}\ ]\!]$ is $[\![\ \langle \mathsf{x}+\mathsf{y} \mid \mathsf{y}{:=}\mathsf{x}\cdot\mathsf{z}\rangle\ ]\!]$ and $[\![\ \mathcal{B}\ ]\!]$ is $[\![\ \mathsf{x}+2\ ]\!]$.

b $[\![\ \mathcal{A}\ ]\!]$ is $[\![\ \langle \mathsf{x}+\mathsf{y} \mid \mathsf{y}{:=}\mathsf{x}\cdot\mathsf{z}\rangle\ ]\!]$ and $[\![\ \mathcal{B}\ ]\!]$ is $[\![\ \mathsf{y}+2\ ]\!]$.

c $[\![\ \mathcal{A}\ ]\!]$ is $[\![\ \langle \mathsf{x}+\mathsf{y} \mid \mathsf{y}{:=}\mathsf{x}\cdot\mathsf{z}\rangle\ ]\!]$ and $[\![\ \mathcal{B}\ ]\!]$ is $[\![\ \mathsf{z}+2\ ]\!]$.

d $[\![\ \mathcal{A}\ ]\!]$ is $[\![\ \langle \mathsf{x}+\mathsf{y} \mid \mathsf{y}{:=}\mathsf{x}\cdot\mathsf{z}\rangle\ ]\!]$ and $[\![\ \mathcal{B}\ ]\!]$ is $[\![\ \mathrm{if}(\mathsf{x}+\mathsf{y},\mathsf{y},4)\ ]\!]$.

7.19 Distribution

In Section 7.5 I stated a number of algebraic rules:

$[\![$ **Mac rule SubXPlusY** : $\langle \mathcal{A}+\mathcal{B} \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S}\rangle + \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S}\rangle\ ]\!]$

$[\![$ **Mac rule SubXTimesY** : $\langle \mathcal{A}\cdot\mathcal{B} \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S}\rangle \cdot \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S}\rangle\ ]\!]$

$[\![$ **Mac rule SubIf** : $\langle \mathrm{if}(\mathcal{A},\mathcal{B},\mathcal{C}) \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv$
$\mathrm{if}(\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S}\rangle, \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S}\rangle, \langle \mathcal{C} \mid \mathsf{x}{:=}\mathcal{S}\rangle)\ ]\!]$

$[\![$ **Mac rule SubT** : $\langle \mathsf{T} \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \mathsf{T}\ ]\!]$

$[\![$ **Mac rule SubF'ThreeOfXEnd** : $\langle f_3(\mathcal{A}) \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv f_3(\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S}\rangle\ ]\!]$

free for

Because of the first rule I will say that substitution *distributes* over addition: You may do addition first and then substitution as on the left side of the rule, or you may do substitution first and then addition as on the right side.

Does substitution distribute over everything? Well—almost. Substitution distributes over all constructs I have presented so far except variables and substitution itself. I stated the rules for variables in Section 7.7:

[**Mac rule SubXX** : $\langle \mathsf{x} \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \mathcal{S}$]

[**Mac rule SubXY** : **if** distinct(x,y) **then** $\langle \mathsf{y} \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \mathsf{y}$]

The following is certainly false:

$$[\![\, \langle\langle \mathcal{A} \mid \mathcal{B}{:=}\mathcal{C}\rangle \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \langle\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{S}\rangle \mid \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S}\rangle{:=}\langle \mathcal{C} \mid \mathsf{x}{:=}\mathcal{S}\rangle\rangle \,]\!]^{\circ}.$$

The rule is not even syntactically valid since $[\![\, \langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S}\rangle \,]\!]$ is not a variable. Hence, substitution does not distribute over substitution.

I will say that $[\![\, \langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle \,]\!]$ is a *binding construct* because it binds a variable. In general, substitution does not distribute over binding constructs.

Constructs you define with $[\![\, \doteq \,]\!]^{\circ}$ and $[\![\, \doteqdot \,]\!]^{\circ}$ may be binding or non-binding. As an example,

$$[\![\, \text{set-to-seven}(\mathsf{x}, \mathsf{y}) \doteq \langle \mathsf{y} \mid \mathsf{x}{:=}7\rangle \,]\!]$$

defines $[\![\, \text{set-to-seven}(\mathsf{x}, \mathsf{y}) \,]\!]$ as a new, binding construct. $[\![\, \text{set-to-seven}(\mathcal{A}, \mathcal{B}) \,]\!]$ is only syntactically valid when $[\![\, \mathcal{A} \,]\!]$ is a variable. As an example you have

$$[\![\, \text{set-to-seven}(\mathsf{z}, 1 + \mathsf{z} + 2) \equiv 1 + 7 + 2 \equiv 10 \,]\!].$$

Substitution does not distribute over substitution, but the following "semisubstitution" rule holds: If $[\![\, \mathsf{x} \,]\!]$ and $[\![\, \mathsf{y} \,]\!]$ are distinct variables and if $[\![\, \mathsf{x} \,]\!]$ does not occur free in $[\![\, \mathcal{S} \,]\!]$, then $[\![\, \langle\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle \mid \mathsf{y}{:=}\mathcal{S}\rangle \equiv \langle\langle \mathcal{A} \mid \mathsf{y}{:=}\mathcal{S}\rangle \mid \mathsf{x}{:=}\langle \mathcal{B} \mid \mathsf{y}{:=}\mathcal{S}\rangle\rangle \,]\!]$:

[**Mac rule SubXSubYNotFree** : **if** distinct(x,y) $\wedge$ notfree(x,$\mathcal{S}$) **then** $\langle\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle \mid \mathsf{y}{:=}\mathcal{S}\rangle \equiv \langle\langle \mathcal{A} \mid \mathsf{y}{:=}\mathcal{S}\rangle \mid \mathsf{x}{:=}\langle \mathcal{B} \mid \mathsf{y}{:=}\mathcal{S}\rangle\rangle$]

If $[\![\, \mathsf{x} \,]\!]$ is not free in $[\![\, \mathcal{S} \,]\!]$ then $[\![\, \langle \mathcal{S} \mid \mathsf{x}{:=}\mathsf{T}\rangle \equiv \mathcal{S} \,]\!]$. Furthermore, $[\![\, \mathsf{x} \,]\!]$ is not free in $[\![\, \langle \mathcal{S} \mid \mathsf{x}{:=}\mathsf{T}\rangle \,]\!]$ regardless of whether or not $[\![\, \mathsf{x} \,]\!]$ is free in $[\![\, \mathcal{S} \,]\!]$. Hence, if I replace $[\![\, \mathcal{S} \,]\!]$ by $[\![\, \langle \mathcal{S} \mid \mathsf{x}{:=}\mathsf{T}\rangle \,]\!]$ in the rule above, then the new formulation says the same, but I don't need to assume that $[\![\, \mathsf{x} \,]\!]$ is not free in $[\![\, \mathcal{S} \,]\!]$ anymore. That gives a purer, algebraic rule:

[**Mac rule SubXSubY** : **if** distinct(x,y) **then** $\langle\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle \mid \mathsf{y}{:=}\langle \mathcal{S} \mid \mathsf{x}{:=}\mathsf{T}\rangle\rangle \equiv \langle\langle \mathcal{A} \mid \mathsf{y}{:=}\langle \mathcal{S} \mid \mathsf{x}{:=}\mathsf{T}\rangle\rangle \mid \mathsf{x}{:=}\langle \mathcal{B} \mid \mathsf{y}{:=}\langle \mathcal{S} \mid \mathsf{x}{:=}\mathsf{T}\rangle\rangle\rangle$]

Another rule of interest reads

[**Mac rule SubXSubX** : $\langle\langle \mathcal{A} \mid \mathsf{x}{:=}\mathcal{B}\rangle \mid \mathsf{x}{:=}\mathcal{S}\rangle \equiv \langle \mathcal{A} \mid \mathsf{x}{:=}\langle \mathcal{B} \mid \mathsf{x}{:=}\mathcal{S}\rangle\rangle$]

---

distribute
binding construct

7.20 Automatic renaming

The macro definition [plus-four(x) ≝ ⟨x + y | y:=4⟩] tells Map that any occurrence of [plus-four(x)] is shorthand for [⟨x + z | z:=4⟩] where it is up to Map to choose a variable [z] that does not conflict with other variables.

When computing [⟨plus-four(y) | y:=3⟩], Map first translates [⟨plus-four(y) | y:=3⟩] to e.g. [⟨⟨y + z | z:=4⟩ | y:=3⟩] and then computes [⟨⟨y + z | z:=4⟩ | y:=3⟩] to [7].

7.21 The meaning of free variables

A free variable in an expression refers to something outside of the expression. As an example you can say "if [x ≡ F] then [if(x, 0, 1) ≡ 1]". [x] is free in [if(x, 0, 1) ≡ 1], so [x] refers to something outside [if(x, 0, 1) ≡ 1]. In this case, [x] is [F] as stated by [x ≡ F].

On the other hand, "if [y ≡ 2] then [⟨y + 1 | y:=7⟩ ≡ ⟨2 + 1 | y:=7⟩]" is wrong. The leftmost [y] in [⟨y + 1 | y:=7⟩] is bound and does not refer outside the expression. "If [y ≡ 2] then [⟨y + 1 | y:=7⟩ ≡ ⟨2 + 1 | 2:=7⟩]°" is even worse since [$\mathcal{B}$] in [⟨$\mathcal{A}$ | $\mathcal{B}$:=$\mathcal{C}$⟩] must be a variable.

If you ask "when does [if(x, 0, 1) ≡ 0] hold?" then the answer could be "when [x] is [T]". This is another use of free variables. Here, [if(x, 0, 1) ≡ 0] is a riddle that can be solved.

When I say [if(T, u, v) ≡ u] with no qualification at all, I tacitly mean that the equation [if(T, u, v) ≡ u] holds for all mathematical objects [u] and [v]. Hence, when an equation like [if(T, u, v) ≡ u] is stated as a fact, lemma, theorem, axiom, or whatever, it is understood that the equation holds in any environment, i.e. regardless of the values of the free variables.

[if(x, T, F) ≡ T] holds for some but not all [x]. Hence [if(x, T, F) ≡ T] stated in isolation fails because it fails for some [x].

7.22 Substitution of equals

If [$\mathcal{A}$ ≡ $\mathcal{B}$] holds for all values of free variables, then [$\mathcal{A}$] and [$\mathcal{B}$] are indistinguishable, and you can replace the one by the other in any expression without affecting the meaning of the expression. This is known as *substitution of equals*, which I mentioned in Section 1.5.

As an example, [if(T, u + v, v) ≡ u + v], so [⟨if(T, u + v, v) + 2 | u:=5⟩ ≡ ⟨u + v + 2 | u:=5⟩]. [v] is free and [u] is bound in [⟨if(T, u + v, v) + 2 | u:=5⟩ ≡ ⟨u + v + 2 | u:=5⟩], so this is an example of substitution of equals where some of the variables are free and some are bound. The validity of substitution of equals relies on the convention that [if(T, u + v, v) ≡ u + v] tacitly means that [if(T, u + v, v) ≡ u + v] holds for all mathematical objects [u] and [v] and, hence, [if(T, u + v, v) ≡ u + v] holds in any context.

7.23 Grand substitution

In Section 7.7 you saw the following lemma and proof:

[Mac lemma L7.7.1 : $\langle x \cdot 3 + x \cdot y \mid x{:=}2 \rangle \equiv 2 \cdot 3 + 2 \cdot y$ **]**

[Mac proof of L7.7.1**:**

Algebra ▷	$\langle x \cdot 3 + x \cdot y \mid x{:=}2 \rangle$	;
Replace ▷ SubXPlusY ▷	$\langle x \cdot 3 \mid x{:=}2 \rangle + \langle x \cdot y \mid x{:=}2 \rangle$	;
Replace ▷ SubXTimesY ▷	$\langle x \mid x{:=}2 \rangle \cdot \langle 3 \mid x{:=}2 \rangle + \langle x \cdot y \mid x{:=}2 \rangle$	;
Replace ▷ SubXX ▷	$2 \cdot \langle 3 \mid x{:=}2 \rangle + \langle x \cdot y \mid x{:=}2 \rangle$	;
Replace ▷ SubNumeral ▷	$2 \cdot 3 + \langle x \cdot y \mid x{:=}2 \rangle$	;
Replace ▷ SubXTimesY ▷	$2 \cdot 3 + \langle x \mid x{:=}2 \rangle \cdot \langle y \mid x{:=}2 \rangle$	;
Replace ▷ SubXX ▷	$2 \cdot 3 + 2 \cdot \langle y \mid x{:=}2 \rangle$	;
Replace ▷ SubXY ▷	$2 \cdot 3 + 2 \cdot y$	]

Proofs like the one above take up a lot of space but say very little. To save space and time, I introduce one, grand substitution rule that allows you to perform substitutions in one step. The rule says that if two expressions $[\,\mathcal{A}\,]$ and $[\,\mathcal{B}\,]$ can be made identical by performing substitutuions and renaming bound variables, then $[\,\mathcal{A} \equiv \mathcal{B}\,]$:

[Mac rule Substitution : **if** Substitution$(\mathcal{A}, \mathcal{B})$ **then** $\mathcal{A} \equiv \mathcal{B}$ **]**

This rule allows to simplify the proof of L7.7.1:

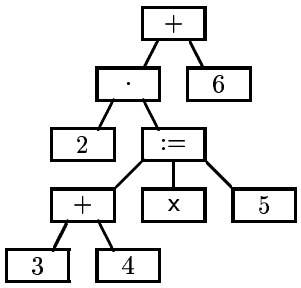
[Mac proof of L7.7.1**:**

Algebra ▷	$\langle x \cdot 3 + x \cdot y \mid x{:=}2 \rangle$	;
Substitution ▷	$2 \cdot 3 + 2 \cdot y$	]

7.24 Answers

Answer 7.1.1 (a) $[\,10\,]$, and (b) $[\,12\,]$.

Answer 7.2.1



Answer 7.4.1 $[\,9\,]$.

Answer 7.7.2

[Mac proof of L7.7.3**:**

Algebra ▷	$\langle \mathsf{if}(\mathsf{x},\mathsf{y},\mathsf{z}) \mid \mathsf{x}{:=}\mathsf{T}\rangle$	;
Replace ▷ SubIf ▷	$\mathsf{if}(\langle \mathsf{x} \mid \mathsf{x}{:=}\mathsf{T}\rangle, \langle \mathsf{y} \mid \mathsf{x}{:=}\mathsf{T}\rangle, \langle \mathsf{z} \mid \mathsf{x}{:=}\mathsf{T}\rangle)$	;
Replace ▷ SubXX ▷	$\mathsf{if}(\mathsf{T}, \langle \mathsf{y} \mid \mathsf{x}{:=}\mathsf{T}\rangle, \langle \mathsf{z} \mid \mathsf{x}{:=}\mathsf{T}\rangle)$	;
Replace ▷ IfTrue ▷	$\langle \mathsf{y} \mid \mathsf{x}{:=}\mathsf{T}\rangle$	;
Replace ▷ SubXY ▷	y	**]**

Answer 7.8.1

a $\lceil\!\lceil \langle \mathsf{x}\cdot\mathsf{x} \mid \mathsf{x}{:=}2+2\rangle + \langle \mathsf{x}+\mathsf{x} \mid \mathsf{x}{:=}1\rangle \rceil\!\rceil$.

b $\lceil\!\lceil \langle \mathsf{x} \mid \mathsf{x}{:=}2\rangle + \langle \mathsf{x} \mid \mathsf{x}{:=}3\rangle + \langle \mathsf{x} \mid \mathsf{x}{:=}4\rangle \rceil\!\rceil$.

Answer 7.10.1 The variables are in order: a bound $\lceil\!\lceil \mathsf{x} \rceil\!\rceil$, a free $\lceil\!\lceil \mathsf{y} \rceil\!\rceil$, a binding $\lceil\!\lceil \mathsf{x} \rceil\!\rceil$, a free $\lceil\!\lceil \mathsf{x} \rceil\!\rceil$, a free $\lceil\!\lceil \mathsf{x} \rceil\!\rceil$, a free $\lceil\!\lceil \mathsf{x} \rceil\!\rceil$, a bound $\lceil\!\lceil \mathsf{y} \rceil\!\rceil$ and a binding $\lceil\!\lceil \mathsf{y} \rceil\!\rceil$.

Answer 7.11.1 (a) and (c) are structurally equal.

Answer 7.13.1 $\lceil\!\lceil \mathsf{x} \rceil\!\rceil$ can be replaced by any variable except $\lceil\!\lceil \mathsf{v} \rceil\!\rceil$. $\lceil\!\lceil \mathsf{y} \rceil\!\rceil$ can be replaced by any variable except $\lceil\!\lceil \mathsf{z} \rceil\!\rceil$ and $\lceil\!\lceil \mathsf{u} \rceil\!\rceil$.

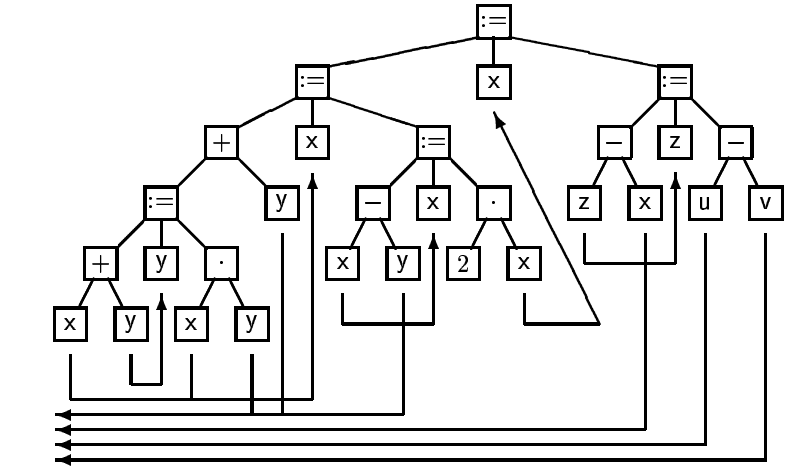
Answer 7.14.1 (a) $\lceil\!\lceil 6 \rceil\!\rceil$, (b) $\lceil\!\lceil (2+\mathsf{x})\cdot\mathsf{y} \rceil\!\rceil$, and (c) $\lceil\!\lceil 7 \rceil\!\rceil$.

Answer 7.15.1

a $\lceil\!\lceil \langle\langle \mathsf{u}\cdot\mathsf{v} \mid \mathsf{u}{:=}\mathsf{v}\rangle \mid \mathsf{v}{:=}\mathsf{x} + \langle \mathsf{w}\cdot\mathsf{y} \mid \mathsf{w}{:=}5\rangle\rangle \rceil\!\rceil$.

b $\lceil\!\lceil \langle\langle \mathsf{u}-1 \mid \mathsf{u}{:=}\mathsf{v}\cdot\mathsf{v}\rangle + \mathsf{v} \mid \mathsf{v}{:=}\mathsf{x}-1\rangle \rceil\!\rceil$.

Answer 7.16.1



Answer 7.18.1 $\lceil\!\lceil \mathcal{B} \rceil\!\rceil$ is free for $\lceil\!\lceil \mathsf{x} \rceil\!\rceil$ in $\lceil\!\lceil \mathcal{A} \rceil\!\rceil$ in case (a) and (c).